GÉRI NATALINO DUTRA

Caracterização de Parâmetros para Modelos de Serviço HTTP

Dissertação de Mestrado apresentada ao Curso de Mestrado em informática, do Departamento de Informática, Setor de Ciências Exatas, Universidade Federal do Paraná, como requisito parcial para a obtenção do título de Mestre em Informática.

Orientadora: Dr.ª Cristina Duarte Murta

CURITIBA
2004

Ministério da Educação
Universidade Federal do Paraná
Mestrado em Informática

PARECER

Nós, abaixo assinados, membros da Banca Examinadora da defesa de Dissertação de Mestrado em Informática, do aluno Géri Natalino Dutra, avaliamos o trabalho intitulado, "*CARACTERIZAÇÃO DE PARÂMETROS PARA MODELOS DE SERVIÇO HTTP*", cuja defesa foi realizada no dia 30 de abril de 2004, às dez horas, no Auditório do Departamento de Informática da Universidade Federal do Paraná. Após a avaliação, decidimos pela aprovação do candidato.

Curitiba, 30 de abril de 2004.

Profª. Dra. Cristina Duarte Murta
DINF/UFPR - Orientadora

Profª. Dra. Keiko Verônica Ono Fonseca
CEFET/CPGEI - Membro Externo

Prof. Dr. Roberto André Hexsel
DINF/UFPR – Membro Interno

Agradecimentos

Gostaria de fazer aqui um agradecimento a todos aqueles que de alguma forma tiveram uma contribuição nesta dissertação.

Em primeiro lugar gostaria de agradecer a minha orientadora Cristina, pela confiança que em mim depositou, aceitando orientar esta dissertação, pela sua dedicação, competência e cooperação de extrema importância.

Aos meus pais por sempre me incentivarem a estudar e a batalhar por um futuro melhor. Também devo a vocês a existência de pessoas tão maravilhosas como os meus irmãos, os quais me deram lições de vida valiosas, sempre me apoiaram e entenderam o meu afastamento para estudar, agradeço a vocês de coração.

Aos meus amigos por torcerem por mim, por entenderem o quão importante este trabalho era. Por rezarem pelo meu sucesso e demonstrarem tanta felicidade quanto à sentida por mim nos momentos em que os resultados iam sendo obtidos, estes momentos estão guardados na minha mente e no meu coração.

Aos meus colegas de trabalho por entenderem a minha ausência, pelo companheirismo e senso de equipe. Uma citação especial para a minha protetora Ivone, a qual apostou e sempre confiou cegamente em mim, devo grande parte deste trabalho a ti e a tua família.

A minha esposa Marli, pelo amor, pela companhia, bom senso, inteligência e por ter me dado o maior presente que já ganhei, o meu filho Luís Fernando, que me fez perceber como a vida é maravilhosa.

Resumo

Uma caracterização contínua e atual da carga da Web é fundamental para a execução de simulações, para a aplicação de modelos analíticos e de planejamento de capacidade. Embora modelos de tráfego Web tenham sido propostos anteriormente, é importante renovar e atualizar estes modelos continuamente, com novos dados, de forma a refletir a evolução dos sistemas, protocolos e do uso da rede. Esta evolução implica em mudanças na carga que devem ser acompanhadas.

Esta dissertação apresenta uma caracterização de uma carga Web recente, focalizando quatro características: o tempo de serviço de uma requisição HTTP, o tamanho da resposta à requisição, a correlação entre o tempo de serviço e o tamanho da resposta e a taxa de serviço dos servidores HTTP. Os resultados destas caracterizações servem para auxiliar a estimativa de valores de parâmetros em modelos de carga. Outro aspecto explorado é a comparação e análise da evolução deste tipo de carga, em relação aos aspectos caracterizados, ao longo dos últimos anos. Os resultados desta caracterização podem ser utilizados em modelos analíticos e em projetos de geradores de carga.

Abstract

A contemporary characterization of the Web workload is fundamental for driving simulations, analysis and capacity planning models. While previous models of Web traffic have been presented in the literature, it is important to renew and update these models continuously with new data to keep up with the evolution of the systems, protocols and network usage, which may trigger changes in the workload. The Web is a continuous evolving system, and to understand the changes occurring in the Web workload, an analysis of the Web traffic over time must also to be continuously undertaken.

This dissertation presents a characterization of a recent Web workload. This study focuses on four characteristics: the service time of an HTTP request, the size of the response of an HTTP request, the correlation between the response size and the service time, and the throughput of the HTTP servers. The characterization results helps to estimate parameter values for workload models. Another issue that we explore in this work is the comparison and analysis of the evolution of the workload registered during the last few years.

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

O desempenho dos sistemas computacionais depende fundamentalmente das características da carga. Assim, um dos primeiros passos em um estudo de avaliação de desempenho é entender e caracterizar a carga de trabalho. Todas as abordagens para avaliação de desempenho, seja avaliação experimental, simulação ou modelagem analítica, requerem caracterização de carga.

Assim como os sistemas, as cargas de trabalho são entidades dinâmicas, que evoluem com o tempo. As cargas de trabalho são geradas, em última análise, por pessoas, usuários dos sistemas. As pessoas tendem a alterar suas rotinas em função de facilidades ou dificuldades que encontram na interação com o sistema, em função de novas descobertas e novos serviços. Estas alterações implicam em mudanças nas cargas dos sistemas.

Um exemplo desta evolução é a Internet. Nos seus mais de trinta anos de existência, esta rede já experimentou mudanças significativas em sua carga, em termos de quantidade e composição. À medida em que a interação fica mais fácil e mais rápida, e novos serviços são oferecidos, os usuários vão se adaptando a estas novas condições e gerando carga que retrata estas mudanças. Portanto, a carga de um sistema de interesse para avaliação e projeto de desempenho deve ser caracterizada não apenas uma vez, mas deve ser estudada de forma contínua.

No caso da Web, a caracterização da carga e do tráfego é fundamental para as atividades de modelagem, projeto e avaliação do desempenho dos sistemas Web, sejam servidores, caches, redes, protocolos ou componentes de programas. Com relação à Internet, a coleta das medições permite melhor compreensão da dinâmica do tráfego e do desempenho da rede. Além disso,

conhecer o tráfego e a carga pode beneficiar também as atividades de operação da rede, de criação de conteúdo, o provimento de acesso e de hospedagem de páginas e domínios [26].

O número de usuários da Web é significativamente maior a cada ano. O número de páginas com recursos multimídia também aumenta e, conseqüentemente, mais arquivos com sons, imagens e vídeos estão disponíveis nas páginas Web. Novas aplicações e protocolos surgem, assim como novas implementações de protocolos, novos meios de acesso à rede. Todas estas mudanças refletem em alterações na carga e no desempenho dos sistemas, e reforçam a necessidade de avaliação continuada da carga.

Este trabalho trata de aspectos específicos da caracterização de carga da Web e tem dois objetivos. O primeiro é dar continuidade a esforços anteriores de caracterização de carga. O segundo objetivo é modelar um dos aspectos da carga ainda não modelado, os tempos de serviço nos servidores HTTP.

A carga da Web tem sido estudada desde seu surgimento. Vários trabalhos foram publicados nos últimos dez anos [1, 2, 4, 5, 9, 15]. A caracterização de carga recente da Web permite a comparação e a avaliação das mudanças ocorridas nas características deste tráfego. Esta caracterização é util também para auxiliar a estimativa de valores de parâmetros em modelos de carga, bem como para o estudo da evolução da carga através da construção de uma série histórica de características e valores.

Um modelo de carga é utilizado para gerar uma carga de trabalho sintética, que pode ser usada com o objetivo de testar um sistema em um ambiente controlado. Um estudo detalhado da carga de trabalho pode ser utilizado para a criação de modelos de carga que preservem as características mais relevantes da carga real. A utilização de modelos de carga é bastante ampla. Modelos de carga são indispensáveis para a modelagem analítica de desempenho e são muito utilizados em simulações. Uma vez que o modelo está disponível, é possível alterar seus parâmetros para refletir mudanças no sistema ou na carga de trabalho real.

A carga de trabalho considerada neste estudo consiste em conjuntos de requisições Web registradas a partir de servidores *proxy-cache* [29]. Esta carga é composta basicamente de requisições de leitura de arquivos feitas através do protocolo HTTP. Estes arquivos são requisitados e exibidos pelos navegadores (*browsers*) e os formatos mais comuns são `.html`, `.gif`, `.jpg`, `.txt`,

dentre outros.

Um servidor *proxy-cache* opera tanto como um cliente quanto como um servidor Web. Este tipo de servidor é localizado em pontos específicos da rede, entre clientes e servidores. Em particular, os servidores cache cuja carga foi analisada neste trabalho são servidores localizados na Internet 2 americana, que tratam milhões de requisições diariamente. Esta localização é estratégica, pois passam por estes caches requisições de usuários de todo o mundo, feitas para servidores cuja distribuição geográfica é bastante diversa. Assim, podemos considerar que estas cargas refletem o comportamento de toda a Internet, pois não dependem de um conjunto específico de clientes ou de servidores.

Um dos aspectos fundamentais desta carga, descrita como um conjunto de requisições de arquivos, é o tamanho dos arquivos servidos. Esta característica define a quantidade de tráfego na rede e sua utilização. Os arquivos servidos são denominados "respostas" às requisições Web. O tamanho de uma dada resposta é medido em bytes e corresponde ao número de bytes enviado do servidor para o cliente em resposta à requisição HTTP. A modelagem dos tamanhos das respostas já foi abordada em vários trabalhos [1, 5, 9, 15]. O comportamento da distribuição de cauda pesada (*heavy-tailed distribution*) dos tamanhos das respostas é amplamente reconhecido na literatura.

Cada requisição a um servidor Web ou servidor cache gera uma quantidade de bytes como resposta. O tempo necessário ao servidor para servir esta resposta é denominado *tempo de serviço*. O modelo de comportamento dos tempos de serviço é provavelmente o mais importante para as atividades de projeto, análise e avaliação de desempenho. A descrição do comportamento dos tempos de serviço é imprescindível para a construção de modelos de desempenho analíticos e de simulação.

Apesar desta importância, os tempos de serviço dos servidores cache ainda não foram modelados. Na falta de uma descrição do comportamento dos tempos de serviço, os trabalhos sobre políticas de escalonamento em caches e em servidores [16, 35] e os trabalhos sobre políticas de distribuição de tarefas em conjuntos de servidores [8, 22] assumem que *a duração da transmissão é igual ao tamanho do arquivo transmitido.* No entanto, embora esta assunção seja razoavelmente intuitiva e pouco discutida, a diversidade de tipos de rede e sistemas cliente, os efeitos

do congestionamento das redes tais como perdas de pacotes, a heterogeneidade das condições de conectividade dos clientes dentre outros fatores, podem invalidá-la.

A correlação entre os tamanhos das respostas e os tempos de serviço foi avaliada em dois trabalhos [34, 42] que, no entanto, não apresentaram modelos para o tempo de serviço. Assim, um dos objetivos deste trabalho é propor um modelo para os tempos de serviço. O conhecimento de um modelo de tempo de serviço HTTP tem, pelo menos, duas aplicações principais. A primeira é sua utilização nos modelos de desempenho, sejam eles analíticos ou de simulação. A segunda aplicação é relativa à interação entre os sistemas de cache e a rede. Sistemas de caches, tais como as demais aplicações de redes, devem cooperar com a rede tendo como objetivo prover a estabilidade necessária à rede e manter a qualidade da aplicação. Entender o comportamento dos tempos de serviço dos caches e sua relação intrínseca com o comportamento da rede pode auxiliar o projeto de sistemas de cache que reconheçam as condições da rede e se adaptem a elas.

A principal contribuição deste trabalho é a caracterização dos tempos de serviço em sistemas de cache Web. A correlação entre os tempos de serviço e os tamanhos das respostas é apresentada e discutida em vários aspectos. A abordagem escolhida para obter os resultados propostos é o desenvolvimento de um modelo de tempo de serviço HTTP. O modelo, após validado, poderá ser usado em projetos de desempenho. Outro objetivo é estudar a evolução da carga nos últimos anos, comparando os resultados de caracterização obtidos com resultados já publicados na literatura nos anos anteriores.

Como principais resultados gerados por esta dissertação podemos citar a apresentação de modelos com parâmetros distintos para os tempos de serviço e tamanhos das respostas. Apresentamos também uma caracterização das cargas de trabalho Web com informações recentes e uma análise da evolução desta carga nos últimos anos. As taxas de serviço observadas na carga de trabalho foram também analisadas.

Este trabalho está organizado em capítulos. No segundo capítulo são apresentados os sistemas de cache Web, seu funcionamento, aspectos da sua utilização, as vantagens e desvantagens do seu uso. Os trabalhos relacionados são também discutidos neste capítulo. A carga de trabalho analisada neste trabalho é descrita no terceiro capítulo, uma comparação com os dados reporta-

dos na literatura é apresentada. No quarto capítulo discutimos a correlação entre os tempos de serviço e os tamanhos das respostas. No quinto capítulo são analisados os modelos estatísticos para o tamanho e o tempo de serviço e são propostos novos modelos estatísticos para os tempos de serviço. As taxas de serviço nos servidores cache são apresentadas no capítulo 6. No capítulo 7 são apresentadas as conclusões e as perspectivas de trabalhos futuros. No Apêndice A são descritas as principais características do *Squid*.

Capítulo 2

Sistemas de Cache Web: Conceitos e Trabalhos Relacionados

A partir de 1994, os estudos sobre *caching* na Web foram intensificados, oferecendo alternativas para minimizar alguns dos problemas relacionados ao desempenho da Web. Web *caching* tornou-se uma solução atrativa porque propicia uma redução efetiva da utilização de banda, aumenta a disponibilidade dos documentos e reduz a latência da rede [11].

Neste capítulo apresentamos uma descrição dos sistemas de cache Web, bem como uma comparação destes sistemas com os sistemas de cache utilizados em arquitetura e sistemas operacionais. Os trabalhos relacionados a este são também discutidos neste capítulo.

2.1 Função dos Sistemas de Cache

Os sistemas de cache de memória presentes nos computadores atuais reduzem o tempo médio de acesso à memória principal, melhorando o desempenho dos computadores. A utilização de caches é uma técnica difundida que melhora a escalabilidade e desempenho de sistemas cliente-servidor, pois contribui para diminuir os pontos de contenção na rede ou no servidor, aumentando o número de clientes que podem ser atendidos. Por tratar-se de um sistema cliente-servidor, a Web pode fazer uso de cache, armazenando os objetos localmente ou tão perto do cliente quanto possível. Nesta seção são abordados os sistemas de caches tradicionais e caches Web e suas

principais funções.

2.1.1 Função dos Caches Tradicionais

Quando a memória cache surgiu nos computadores monoprocessados, a preocupação com uso de caches não era tão evidente devido ao fato de o processador funcionar praticamente na mesma velocidade da memória principal. Com o surgimento de processadores mais velozes, a diferença entre a memória principal e o processador tornou-se bastante acentuada, comprometendo o desempenho de novas gerações de computadores. Surgiu então, a idéia de inserir uma pequena quantidade de memória, porém mais veloz, entre o processador e a memória principal para melhorar o tempo de acesso à memória. Essa pequena quantidade de memória passou a ser denominada memória cache, ou simplesmente cache [44]. A memória cache foi inicialmente instalada na placa mãe. Um sistema de memória típico possuía uma única memória cache. Posteriormente, os processadores ganharam uma pequena quantidade de cache. Isto se deu porque a incorporação de cache no processador era cara e, por isso, foi implementada em pequena quantidade [47].

O cache armazena os dados mais requisitados pelo processador, com isso elimina-se a necessidade de buscar ou escrever dados com muita freqüência na memória RAM (*Random Access Memory*), que é mais lenta que o processador e o cache. Assim, evita-se a perda de desempenho da máquina. O cache de memória é usado como um intermediário entre o processador e a memória RAM. Inicialmente a informação é carregada na memória RAM e à medida que os dados são requisitados, estes são armazenados na memória cache. A utilização de caches em arquitetura de computadores e sistemas operacionais é bastante eficaz na redução do tempo médio de acesso à informação [38].

A observação de que referências à memória feitas em qualquer intervalo curto de tempo tendem a usar apenas uma pequena fração da memória total é chamado *princípio da localidade* e forma a base de todos os sistemas de cache [45]. A idéia geral é que, quando uma palavra é referenciada, ela é trazida da grande memória lenta para o cache, de modo que, da próxima vez que for utilizada, pode ser acessada rapidamente. O uso de memória cache é viável devido a dois princípios de acesso à memória: localidade temporal e localidade espacial. A localidade temporal

refere-se ao modelo de referência temporal de um item e indica que, se um item é referenciado, é provável que seja referenciado novamente em um curto espaço de tempo. Sistemas cujas cargas de trabalho apresentem boa localidade de referência temporal podem se beneficiar do uso de estratégias de cache. Localidade espacial refere-se ao modelo de referência relativo ao espaço físico de armazenamento de um item, e indica que, se um item é referenciado, é provável que as posições próximas a este item também o sejam. Sistemas que apresentam esta característica na carga de trabalho podem se favorecer do uso de técnicas de busca antecipada, que armazenam antecipadamente prováveis futuras requisições no cache, com o objetivo de aumentar as taxas de acerto. Todos os computadores comerciais desenvolvidos hoje, desde os mais rápidos até os mais lentos, incluem memória cache [38, 39].

2.1.2 Função dos Caches Web

Os caches Web surgiram com o mesmo princípio dos caches tradicionais: reduzir o tráfego na rede e melhorar o tempo de serviço para os usuários. Assim como o cache na hierarquia de memória, o Web cache armazena os objetos mais acessados em uma área específica para posterior recuperação. Sistemas de cache da Web seguem o mesmo princípio dos sistemas de cache utilizados em arquitetura e sistemas operacionais. A utilização de cache na Web tem o objetivo de melhorar o desempenho na obtenção de respostas.

Servidores cache são apenas um exemplo de uma classe de servidores mais ampla, denominada servidores *proxy* [29]. A característica comum a esses servidores é que eles executam os acessos aos servidores Web finais em nome dos clientes. Com a utilização de *proxies*, as consultas dos clientes Web, que antes eram feitas diretamente para os servidores, passam a ser direcionadas para estes novos servidores. Assim, os servidores *proxy* tornam-se concentradores de requisições e respostas. Portanto, são locais ideais para a implementação de caches.

Os *proxies* podem estar associados a caches ou não. Um *proxy* não associado a cache simplesmente encaminha pedidos e respostas. Um *proxy* com *caching* trabalha como cliente e como servidor. Ao receber requisições dos clientes, atua como servidor, e quando faz requisições ao servidor Web, atua como cliente. O *proxy* intercepta as requisições HTTP dos clientes, verifica se o objeto requerido está no cache. Caso o encontre, retorna o objeto ao cliente. Se o objeto

não é encontrado, faz a solicitação ao servidor Web, e armazena uma cópia no cache e, então, retorna o objeto ao cliente.

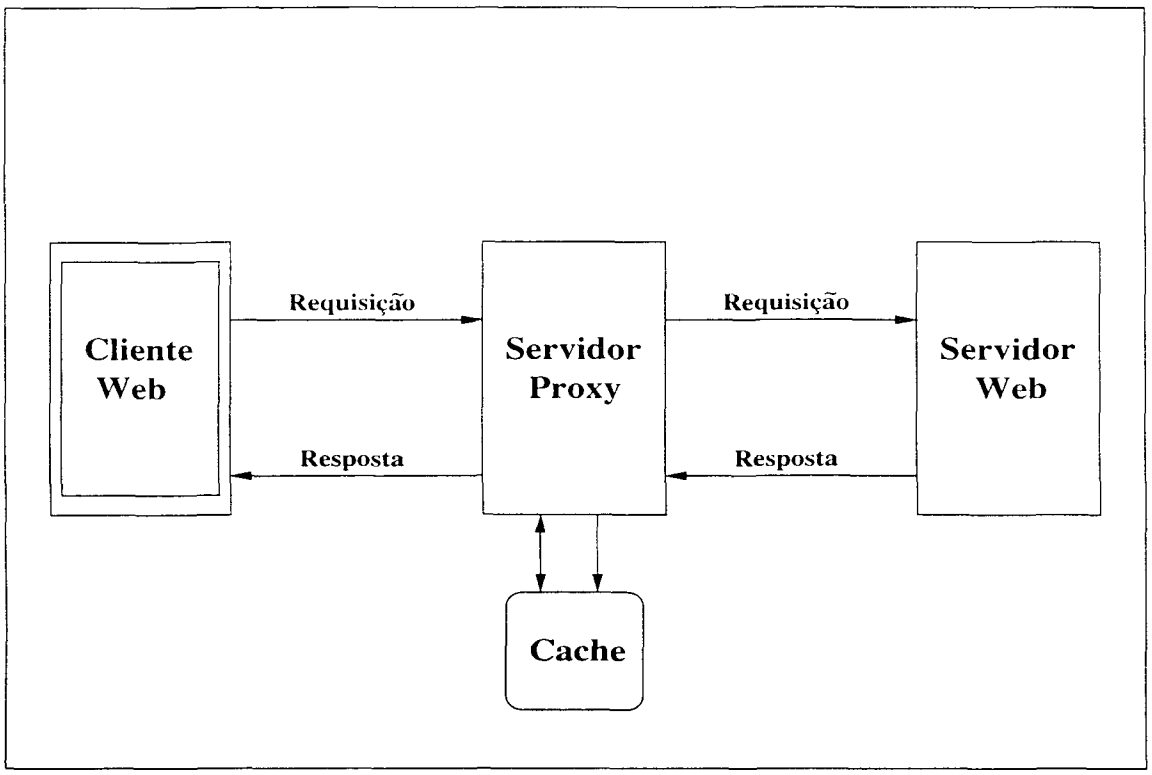


Figura 2.1: Interação do cache com o cliente e o servidor Web através do servidor *proxy*.

Como mostra a Figura 2.1, os clientes Web, através dos navegadores da Internet (*browsers*), fazem conexão com os servidores remotos. Entretanto, os navegadores podem ser configurados para conectarem-se a um servidor cache, que pode ser implementado em um *proxy*. Assim, quando um usuário solicita uma página, o navegador primeiramente verifica seu cache local e, se o recurso (URL - *Universal Resource Locator*) não é encontrado, o navegador envia a solicitação para o servidor *proxy* local. Se o *proxy* local tem a cópia da página requisitada (cache *hit*) e essa cópia não expirou, o *proxy* local retorna a página imediatamente. Caso a página não seja encontrada (cache *miss*) ou essa expirou, o *proxy* local a buscará então no servidor remoto e, ao receber a resposta, manterá uma cópia em seu cache enviando uma cópia para o usuário.

2.1.3 Comparação entre Caches Tradicionais e Caches Web

Os caches da Web tem muitas semelhanças com os caches tradicionais, principalmente no que se refere à função. No entanto, há algumas diferenças entre eles, as principais são:

- Tamanhos dos objetos: a unidade básica de transferência para o cache Web é o arquivo; os caches para Web armazenam integralmente os arquivos transferidos, sendo que estes arquivos têm tamanhos variávies. O cache tradicional é dividido em blocos do mesmo tamanho, sendo que a unidade de transferência e armazenamento entre este e a memória principal é o bloco. A operação com objetos de mesmo tamanho ou de tamanhos bem diversos tem importantes implicações nas políticas de substituição de objetos no cache e também no armazenamento;

- Variação da carga de trabalho: as cargas dos dois sistemas são bastante diferentes. Enquanto os caches tradicionais tem uma carga local e com blocos de tamanho único, os caches Web tem uma carga composta por milhões de arquivos distribuídos geograficamente com características de acesso e armazenamento bem distintas;

- Possibilidades de operação de escrita: os caches para a Web não habilitam a operação de escrita, apenas leitura. Os clientes não gravam informações no cache. Os caches tradicionais executam a operação de escrita e estes dados são repassados, imediatamente ou não, à memória principal;

- Latência: nos caches Web o tempo de serviço varia mesmo para arquivos do mesmo tamanho, pois alguns fatores como largura de banda, congestionamento da rede, características do servidor, características do cliente, a distância física entre o cache e a origem do dado influenciam a resposta. Em caches tradicionais, o tempo de espera é praticamente fixo e medido em ciclos de *clock*. Os tempos de *miss* são muito mais homogêneos nos caches tradicionais.

2.2 Tipos de Caches Web

Como os caches de memória e de disco, o cache Web armazena os dados em uma área específica para posterior recuperação. Os caches podem ser implementados nos vários componentes do sistema e podem ser classificados de acordo com a sua localização. A Figura 2.2 mostra as diferentes localizações dos caches. Entre os caches de clientes, implementados no *browser*, e os caches de servidor, implementados no próprio servidor, há caches de rede, implementados em pontos estratégicos da rede, e caches reversos, que auxiliam o servidor a manter um nível adequado de carga. O objetivo do cache de rede é atender a um grande conjunto de clientes que utilizam a mesma saída (*backbone*) para a Internet [48].

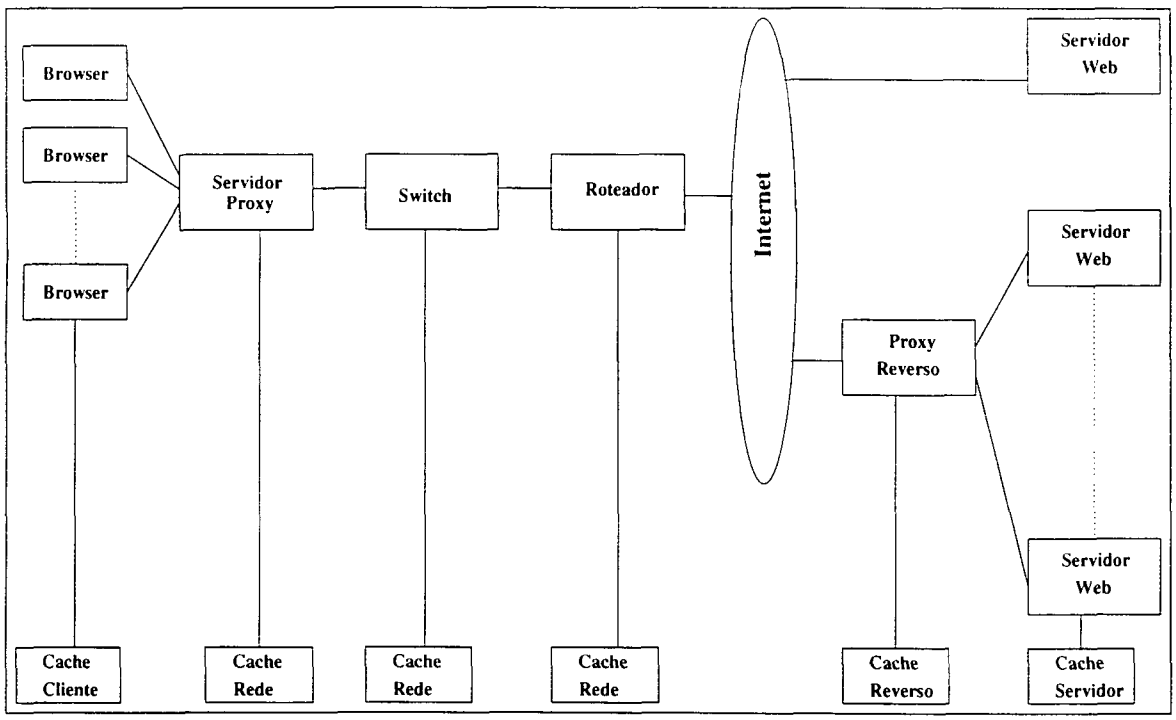


Figura 2.2: Localização dos diferentes tipos de cache.

Os caches de clientes são implementados no *browser*, devido ao fato de que é bastante provável que o usuário volte a acessar as mesmas páginas com freqüência. Assim, os navegadores Web oferecem um cache próprio. Em caso de *hit* evitar-se-á o acesso à rede e o cache proporcionará uma velocidade muito grande na recuperação dos objetos. Este tipo de cache não aproveita os

recursos solicitados pelos demais usuários no mesmo ambiente de rede local.

Os caches de servidor mantêm as informações acessadas com maior freqüência, armazenando-as na memória principal do servidor. Os objetos que poderão estar no cache se resumem aos objetos deste servidor.

Os caches de rede são implementados com o objetivo de atender os clientes que utilizam a mesma saída para a Internet. Esta implementação permite que cópias de todos os objetos acessados pela rede fiquem disponíveis, para que vários usuários tenham acesso a estes objetos. Os caches de rede armazenam requisições de conjuntos diversos de usuários, que podem requisitar páginas de toda a Web. Devido a essa diversidade, possivelmente sua capacidade de armazenamento será preenchida antes do que os caches de clientes, por isso, num servidor poderão ser removidas ou substituidas respostas mais cedo do que nos caches de clientes.

Um *proxy* pode ser transparente, não modificando as mensagens que fluem pelo *proxy*, ou não-transparente, podendo modificar o pedido ou a resposta. No caso mais geral, *proxies* operam como intermediários que podem aplicar transformações ou outras operações especiais sobre os documentos transmitidos [25].

Os caches de *proxy* transparente são usados nos servidores de Internet e não requerem configuração do lado dos clientes. Os usuários não percebem que estão usando um *proxy*. O roteador é programado para que intercepte as conexões HTTP e as redirecione ao servidor *proxy*. A definição mais trivial de *proxy* transparente é que o usuário não perceberá diferenças entre uma requisição feita diretamente ao servidor e uma requisição executada através de servidores *proxy*. Não há necessidade de configurar o navegador do cliente.

Os caches de *proxy* reverso, conforme mostrado na Figura 2.2, estão mais próximos dos servidores, ao contrário do cache de *proxy* que está mais próximo do cliente. Os caches de *proxy* reverso interceptam as requisições destinadas a um ou mais servidores, e têm como função replicar o conteúdo para as diversas áreas geográficas, além de fazer balanceamento de carga. Os caches reversos são independentes dos caches implantados próximos dos clientes, mas podem atuar conjuntamente para melhorar o desempenho da Web [29].

2.2.1 Critérios para o Armazenamento de Objetos nos Caches

Nem todos os objetos Web são armazenados nos servidores cache. Um cache pode decidir se uma resposta deve ser armazenada com base em dois fatores: requisitos relacionados ao protocolo HTTP e os requisitos específicos do conteúdo, que são afetados pelos requisitos de um cache e pelas diretrizes que determinam a freqüência da revalidação do cache. Quanto ao protocolo, deve-se levar em consideração os atributos da mensagem, como tamanho ou tipo de conteúdo, pois estas informações serão avaliadas para identificar a possibilidade de inclusão do arquivo.

Os caches Web precisam implementar as restrições impostas pelo HTTP. Por padrão, a resposta à uma requisição é armazenada no cache se os métodos e campos do cabeçalho da requisição indicarem que ela pode ser armazenada. Estes controles podem ser obtidos através de diretivas internas ao objeto como, por exemplo, o *Cache-Control* que permite que um servidor origem indique a possibilidade de armazenamento dos objetos [21]. A seguir são descritas algumas das diversas diretivas de respostas quanto ao tratamento dado pelo cache:

- *Public:* indica que a resposta pode ser armazenada por qualquer cache;
- *Private:* indica que toda ou parte da resposta é de interesse de um único usuário e não pode ser armazenada por um servidor que compartilha seu conteúdo;
- *No-cache:* se esta diretiva não especifica um *field-name*, então o cache não pode usar a resposta para atender às requisições subseqüentes sem executar uma revalidação no servidor de origem.

Com relação ao conteúdo, alguns fatores influenciam a decisão de armazenamento dos objetos, tais como o tamanho das respostas e se as respostas são dinâmicas ou incluem *cookies*.

Os documentos estáticos são os documentos mais comumente encontrados em caches, por exemplo, páginas Web (arquivos com terminação `.html`, `.htm`), páginas de texto (`.txt`, `.ps`, `.pdf`), arquivos de imagem (`.gif`, `.jpeg`, `.tif`), arquivos de áudio (`.au`, `.wav`, `.mp3`), arquivos de vídeo (`.mpeg`, `.mp2`) e arquivos dinâmicos (`.cgi`, `.pl`, `.perl`).

Se todos os documentos fossem dinâmicos, a efetividade dos caches seria minimizada. No entanto, hoje, apenas uma fração das requisições corresponde a documentos dinâmicos [26].

O armazenamento das informações em cache Web precisa ter espaço suficiente para as várias necessidades do armazenamento em cache, entre elas, o próprio programa, o espaço para *swap* do sistema operacional, o espaço disponível para o cache e o espaço para geração dos *logs* de acessos. A Tabela 2.1 apresenta uma recomendação de tamanhos de caches baseado no número de clientes, retirado de [31]. A necessidade de espaço por usuário diminui com o crescimento do número de usuários porque a probabilidade de utilização simultânea do provedor decresce com o crescimento do número de usuários.

Número de usuários	Tamanho de cache por usuário	Tamanho total do cache
50	10-20 MB	0,5-1 GB
100	10-15 MB	1-1,5 GB
500	3-4 MB	1,5-2 GB
1000	2-4 MB	2-4 GB
2000	1,5-2,5 MB	3-5 GB
3000	1-2 MB	3-6 GB

Tabela 2.1: Tamanho recomendado para cache baseado no número de usuários.

2.3 Métricas de Desempenho

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

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

e

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

A rigor, todas as requisições são atendidas pelo cache mas as *hits* são atendidas a partir de documentos previamente armazenados no cache enquanto as requisições *misses* são atendidas através de requisições subseqüentes ao servidor.

A otimização das duas métricas é vantajosa para usuários e administradores de servidores Web. A redução do número de requisições aos servidores (maior HR) é melhor para os usuários pois oferece menor latência e, diminui o volume do tráfego além de diminuir a carga no servidor destino da conexão. O descongestionamento da rede (maior BHR) é melhor para os administradores pois há menor uso da rede e, influencia fortemente o tempo de serviço, uma vez que as requisições ausentes nos caches poderão ser respondidas mais rapidamente do que quando há sobrecarga na rede. Portanto, o aumento de ambas as métricas, HR e BHR, ajuda a conter o tráfego na rede, evitando desperdício de recursos (largura de banda), além de melhorar o tempo de serviço.

Outra métrica importante para o desempenho dos caches é o tempo de resposta, que é dado por:

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

No ambiente Web, as escalas de tempo são maiores do que nos sistemas de caches tradicionais, que tem tempo de *miss* constante. Nos caches Web, o tempo de *miss* depende do tamanho da requisição, da largura de banda entre o cliente e o servidor, e das condições de carga na rede e no servidor no momento da busca.

2.4 Utilização de Caches na Web

Os sistemas de cache Web possibilitam que benefícios sejam alcançados com o seu uso, mas alguns possíveis problemas podem ocorrer com o uso do cache. Esta seção descreve as vantagens e desvantagens do uso do cache.

2.4.1 Vantagens da Utilização de Caches na Web

Pode-se citar como vantagens do uso de servidores cache a redução do tráfego, a redução da carga nos servidores, a redução da latência e o aumento da acessibilidade aos documentos.

A redução do tráfego ocorre porque o número de requisições que precisa trafegar na Web diminui sensivelmente. A redução da banda utilizada beneficia todos os usuários da Web e diminui os custos. A diminuição do tráfego será proporcional à taxa de acerto atingida. A redução de tráfego também acarretará a diminuição do congestionamento da rede. Com isso, as transferências de arquivos sofrerão redução da perda de pacotes, e a necessidade de retransmissão diminuirá. A perda de pacotes é um dos maiores problemas para a transferência de arquivos [10].

A redução da carga dos servidores ocorre porque um número menor de requisições precisará ser atendida pelo servidor, pois várias requisições serão atendidas pelos caches. Um servidor *proxy* pode reduzir o problema de sobrecarga do servidor. Esta forma de tratamento das requisições possibilita um servidor lidar com mais pedidos de um conjunto diversificado de clientes, evitando uso de seus recursos para servir requisições redundantes. O maior número de pedidos tratados é refletido não apenas na camada de aplicação, mas também na camada de transmissão. Pode haver uma redução no número de conexões TCP que são recusadas porque a fila de espera está cheia, e o tempo de espera será menor para os pedidos pendentes.

O cache possibilita a redução da latência porque as respostas às requisições de objetos que se encontram nos caches são feitas a partir do cache, diminuindo a carga do servidor Web, ou seja, o acesso tende a ter sua velocidade aumentada. A latência é proporcional à proximidade do cache em relação ao cliente. Vale novamente ressaltar que, resolvendo a requisição no cache, o tempo que o cliente gastará para obter seus resultados é reduzido.

A possibilidade de acesso aos objetos é incrementada porque, mesmo em situações que o servidor esteja inoperante ou sobrecarregado, se a página estiver armazenada no cache será possível acessá-la.

A melhora da conectividade de rede é outra vantagem da utilização de cache. As velocidades das redes cresceram rapidamente, se a velocidade da conexão entre o provedor do usuário e a Internet for alta, as respostas são trazidas rapidamente para a extremidade da rede do provedor. No entanto, se a conexão for lenta, a transferência da resposta para a extremidade da rede do

provedor pode dominar a latência percebida pelo usuário. A velocidade da conexão é limitada à velocidade mais baixa encontrada no caminho [31]. Quando um roteador alcança o limite de sua capacidade, ele descarta os pacotes de dados e faz nova requisição causando atrasos. Com o uso de cache, requisições freqüentemente acessadas pelos navegadores são respondidas passando por um número menor de roteadores, reduzindo a perda de pacotes e o tempo de espera.

Embora os preços dos serviços de Internet continuem caindo e as velocidades de redes continuem crescendo, os caches Web sempre serão necessários pelas seguintes razões [18]:

- A largura de banda sempre terá custo que, provavelmente, nunca chegará a zero, mesmo com o incremento da competitividade e o crescimento do mercado. Assim, o cache sempre será um importante dispositivo para diminuir a diferença entre as taxas de serviço observadas por usuários próximos e mais distantes dos *backbones* principais.

- As variações de largura de banda e das latências persistirão devido às variações de localização e de capacidade observadas nas instalações; muitas vezes as restrições financeiras impedem a atualização dos sistemas. O cache pode diminuir estas variações.

- As distâncias das redes estão aumentando e o aumento do número de dispositivos, tais como *firewalls* e *proxies*, utilizados em mecanismos de segurança e privacidade fazem com que o número de *hosts* através dos quais a informação precisa passar aumente, aumentando os tempos de resposta da Web. Os caches podem contribuir para diminuir estes tempos.

- A demanda pela largura de banda continua a crescer, a quantidade de informações tende a aumentar e as pretensões dos usuários para maiores velocidades garantem que a demanda por largura de banda nunca terminará. O uso eficiente de cache auxilia na diminuição do uso da largura de banda.

- Os picos de acesso (*hot spots*) continuarão existindo, e a distribuição adequada da carga pode minimizar este problema quando a demanda for previsível. Caches podem suavizar os picos de acesso, resultando em uma maior eficiência do tráfego.

- Sempre haverá a necessidade de uso de cache pois estes podem tornar mais eficientes os computadores e melhorar a conectividade das redes.

2.4.2 Desvantagens da Utilização de Caches na Web

Os caches também apresentam alguns fatores que podem ser considerados como desvantagens na sua utilização. Um exemplo disso seria a privacidade no *caching*. Os pedidos de recurso de um usuário na Web podem ser usados por um *proxy* para construir um perfil do usuário.

Os direitos autorais podem ser não respeitados com o uso de caches. Uma resposta de um servidor origem poderia ser colocada em caches intermediários ao longo do caminho de resposta, contra as instruções do servidor origem [26].

Outro fator que prejudica a utilização de cache é a utilização de estratégias, pelos servidores, para impedir o armazenamento de documentos no cache, ou seja, utilizar uma técnica que impeça que um recurso que poderia ser armazenado no cache não o seja como, por exemplo, definir o cabeçalho *expires* com uma data passada. Nem todos os servidores origem estão interessados em permitir que o seu conteúdo seja armazenado em cache. Assim, é possível que o conteúdo do cache fique desatualizado. O protocolo HTTP 1.1 oferece um número considerável de opções para o servidor expressar sua preferência sobre a permissão de cache de determinados recursos [21]. As versões anteriores do protocolo não tinham esta flexibilidade.

Outros problemas a serem considerados nos caches são o cache como ponto de contenção na rede, a replicação de objetos e a computação adicional em casos de *misses*, além do menor número de acessos aos servidores de origem.

Se alguns dos componentes do cache estiverem inadequados para o número de usuários, o cache pode se tornar um ponto de contenção no sistema. O acesso ao cache deve ser pelo menos tão eficiente quanto o acesso ao servidor origem.

Quanto à replicação de objetos, conforme apresentado em [18], existem alguns problemas potenciais a serem considerados na utilização de caches. Dentre eles, o mais significativo é a possibilidade de que o conteúdo disponível no servidor seja mais recente do que o armazenado no cache e o cliente então receba a versão desatualizada. Existe um problema de consistência entre os originais nos servidores e suas cópias nos caches.

Quando os objetos não são encontrados no cache (*miss*), o tempo de serviço pode ser maior do que se não houvesse o cache no caminho da requisição. Há tendência de melhorar a latência somente para respostas encontradas no cache, que são freqüentemente requisitadas. Computação

e comunicação adicional são necessárias no caso de não encontrar o recurso solicitado. O tempo médio de respostas para os *hits* chega ser cinco vezes menor do que para os *misses* [40].

Entre os problemas apresentados, também podem ser considerados o menor número de acessos aos servidores de origem, prejudicando a popularidade do *site*, o que pode ser visto como uma grande desvantagem por parte de empresas e organizações.

2.5 Execução de Uma Requisição HTTP

Os serviços Web tem como base uma infra-estrutura de suporte que compreende muitos recursos de hardware diferentes, incluindo estações de trabalho cliente, servidores e subsistemas de armazenamento, redes, balanceadores de carga, entre outros. Vários tipos de processos de programas, incluindo servidores Web, servidores de aplicação, protocolos, e sistemas operacionais compartilham recursos de hardware. O uso compartilhado desses recursos aumenta a disputa e gera filas de espera. Uma requisição gasta parte de seu processamento recebendo serviço em vários recursos, sendo processada, e outra parte esperando pelo serviço nas filas. Nesta seção são apresentadas descrições de como uma requisição Web é tratada em diferentes níveis. Também são descritos modelos de filas de servidores.

2.5.1 O Modelo TCP/IP

Garantir a operação de uma rede de alcance tão vasto e com tecnologias tão variadas não é uma tarefa simples. Para tornar isso possível, toda comunicação através da Web é organizada em camadas. O modelo utilizado pela Web é denominado Pilha TCP/IP em função dos principais protocolos envolvidos [32].

No modelo TCP/IP, a interface com o hardware de rede é feita pela camada de interface de rede, também chamada de camada de enlace ou camada de enlace de dados. O TCP/IP não especifica qualquer protocolo nesta camada, e pode-se usar quase qualquer interface de rede disponível. A camada de rede, também chamada de camada de Internet ou camada redes, fornece a imagem de uma rede virtual de inter-redes. O protocolo IP (*Internet Protocol*) é o protocolo mais importante nesta camada. O IP não fornece confiabilidade, controle de fluxo ou recuperação de erros. A camada de transporte provê mecanismos para a transferência de dados

de uma ponta a outra. Esta camada é responsável pelo fornecimento de um intercâmbio confiável de informações. O principal protocolo da camada de transporte é o TCP (*Transmission Control Protocol*). Outro protocolo da camada de transporte é o UDP (*User Datagram Protocol*).

No topo do modelo encontram-se serviços e aplicações que se utilizam da camada de transporte para a entrega dos dados, tais como a transferência de arquivos (FTP), correio eletrônico (SMTP - *Simple Mail Transfer Protocol*), terminal virtual (TELNET), dentre outras.

A Web é responsável pelo maior uso da Internet [14, 25, 46]. O HTTP é o protocolo de comunicação da Web. A troca de informações entre os clientes (*browsers*) e servidores é feita através de mensagens HTTP. O HTTP trabalha geralmente sobre o protocolo TCP que, por sua vez, executa sobre o IP, que geralmente executa sobre *Ethernet*. As quatro camadas conceituais são construídas sobre uma quinta camada física de hardware. Na Tabela 2.2 são mostradas as camadas do TCP e alguns dos protocolos que atuam em cada camada.

Camada	Protocolos
Aplicação	HTTP, FTP, Telnet, SMTP, DNS
Transporte	TCP, UDP
Rede	IP
Enlace	*Ethernet*, ISDN, PPP, ATM

Tabela 2.2: Modelo conceitual TCP/IP.

2.5.2 Tratamento de Uma Requisição

Uma requisição Web envolve o estabelecimento de uma conexão direta entre cliente e servidor. A troca de pacotes entre um cliente e um servidor para uma transação HTTP por meio de TCP, apresenta diferenças significativas quando avaliadas as diferentes versões do protocolo HTTP. O mecanismo de conexões persistentes, disponível a partir da versão 1.1 do protocolo HTTP, permite reduzir a latência inerente ao estabelecimento de cada conexão [6, 28].

As transferências da requisição e da resposta pela rede sofrem atrasos tanto da requisição do cliente ao servidor, quanto na resposta do servidor ao cliente. Estas latências são mostradas na

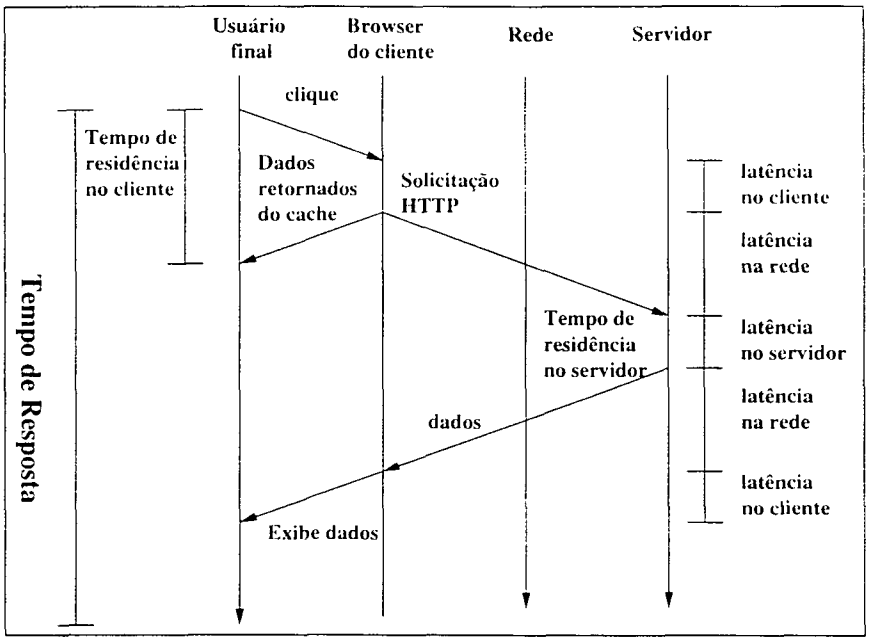


Figura 2.3: Tempos envolvidos na resposta a uma requisição.

Figura 2.3, e ocorrem em função dos diversos componentes na rota entre o cliente e o servidor, como *modems*, roteadores e *switches*. No ponto extremo da solicitação da informação está o servidor. O servidor recebe a solicitação proveniente do cliente, interpreta a solicitação, executa o método solicitado, por exemplo, GET ou POST e, em caso de GET, procura o arquivo que pode estar no cache ou nos discos. O servidor lê o conteúdo em pacotes e os envia para a porta da rede. Quando finalizado o envio, o servidor fecha a conexão. No servidor também há atrasos devido ao processamento das informações, tais como tempo do processador, acesso à placa de interface de rede, ao cache e ao disco. Para obtenção do tempo de resposta são somadas todas as latências e os tempos de serviços. Como já descrito na seção 2.1.2, há possibilidade de intermediários atuarem no caminho entre o usuário e o servidor.

Os servidores Web processam requisições em paralelo, e várias requisições são atendidas concorrentemente. O programa servidor executa continuamente, esperando as requisições dos usuários. As requisições podem ser estáticas, correspondentes a um arquivo lido a partir do sistema de arquivos do servidor, ou dinâmicas, respostas geradas em tempo real em função de parâmetros enviados na requisição.

Tanto para respostas de conteúdo estático como dinâmico, para que as requisições HTTP possam ser transmitidas através do TCP é necessária a abertura de uma conexão. Após a abertura da conexão, o cliente envia a requisição feita pelo usuário, identificada pela URL. Antes de atingir o meio físico de transmissão, um ou mais segmentos TCP são montados e submetidos à camada de rede. Um ou mais datagramas IP são montados e submetidos à interface de rede. Cada datagrama que chega à interface de rede do servidor causa uma interrupção. Esta interrupção é tratada pelo sistema operacional que extrai do datagrama a mensagem HTTP e a envia para a porta dedicada ao servidor Web. Este realiza um processamento sobre a mensagem recebida para saber qual é a informação desejada e verifica, a partir do cabeçalho da mensagem, se ele pode atender a requisição, baseando-se em permissões de acesso, autorização e autenticação. A partir deste ponto começa o processamento da requisição. O servidor procura pelo arquivo solicitado no seu cache em memória e, caso não o encontre, transferências do disco são realizadas. No caso de páginas dinâmicas, os processos que geram os dados requisitados são criados e executados. Uma mensagem HTTP de resposta é então construída e enviada ao usuário. Alguns servidores fazem um registro (*logging*) para cada requisição tratada. Este registro é gravado em arquivo. Se existirem intermediários, eles receptarão as mensagens e, baseado nos processos efetuados, direcionarão a mensagem ao cliente.

2.5.3 Tratamento dos Pacotes de Entrada das Requisições HTTP

Os pacotes contendo as solicitações ou respostas HTTP chegam aleatoriamente na estação destino (servidor ou usuário). Para tratar esta chegada aleatória de pacotes, alguns sistemas utilizam o mecanismo de interrupção de programa. Quando um pacote chega, uma interrupção de hardware ocorre e o *driver* de dispositivo efetua suas tarefas usuais de aceitar o pacote e reinicializar o dispositivo. Antes de retornar da interrupção, o *driver* de dispositivo solicita que o hardware programe a execução de uma segunda interrupção, a qual terá prioridade mais baixa. Essa segunda interrupção, conhecida como interrupção de programa, suspende o processamento e faz a CPU saltar para a rotina que fará o tratamento da interrupção. Desse modo, em alguns sistemas, todo processamento de entrada ocorre como uma série de interrupções [14].

Como a entrada ocorre durante a interrupção, o código do *driver* de dispositivo não pode

chamar procedimentos arbitrários para processar cada pacote: o sistema precisa ser projetado para retornar rapidamente de uma interrupção. Portanto, o procedimento de interrupção não chama o processo do IP diretamente. Além disso, como o sistema utiliza um processo específico para implementar o IP, o *driver* de dispositivo não pode chamar este processo diretamente. Em vez disso, para sincronizar a comunicação, o sistema emprega uma fila em conjunto com primitivas de passagem de mensagens. Quando um pacote que transporta um datagrama IP chega, a rotina de interrupção precisa colocar o pacote em uma fila e passar uma mensagem para comunicar a chegada do pacote ao processo IP. Quando não tem pacotes para tratar, o processo IP fica aguardando a chegada de datagramas. Há uma fila de entrada associada a cada dispositivo de rede; um só processo IP extrai datagramas de todas as filas e os processa.

Ao aceitar um datagrama que chega, o processo IP precisa decidir para onde enviar tal pacote para processamento adicional. Se o datagrama transporta um segmento TCP, o pacote é enviado para o módulo TCP; se transporta um datagrama UDP, o pacote é enviado para o módulo UDP.

Em virtude da complexidade do TCP, a maioria dos sistemas utiliza um processo para gerenciar segmentos TCP. Uma conseqüência da existência de processos IP e TCP separados é que eles precisam usar um mecanismo de comunicação entre processos quando interagem. O IP chama um procedimento para depositar segmentos em uma porta de comunicação e o TCP usa outro procedimento para recuperá-los.

Depois de receber um segmento, o TCP utiliza os números de porta para encontrar a conexão à qual o segmento pertence. Caso o segmento contenha dados, o TCP irá inseri-los em um dispositivo de armazenamento temporário associado à conexão e retornar uma mensagem de confirmação para o transmissor. Caso o pacote que chega transporte uma mensagem de confirmação, o processo de entrada do TCP precisará também se comunicar com o processo de temporização (*timer*) do TCP para cancelar a retransmissão pendente.

Na Figura 2.4 é mostrada a linha de tempo de uma conexão, descrevendo a seqüência de eventos que acontece quando uma requisição do cliente faz uso do cache e uma cópia válida do arquivo solicitado não foi encontrado no cache. O diagrama não mostra as confirmações do TCP, nem a troca de pacotes para fechar a conexão TCP. Através da mesma Figura é possível

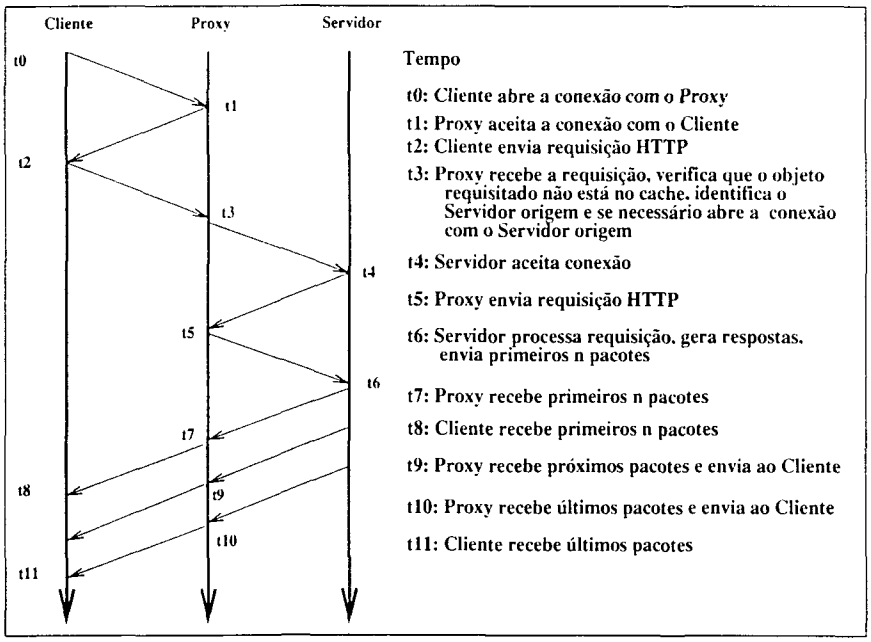


Figura 2.4: Seqüência de eventos para satisfazer uma solicitação do cliente.

verificar os eventos que ocorrem quando uma cópia válida for encontrada no cache, através das ligações entre o cliente e o proxy, e o retorno deste arquivo ao cliente. Os eventos necessários para atender uma requisição que não faz uso de cache podem ser observados através das ligações entre o proxy e o servidor da mesma Figura, pois o proxy está atuando como um cliente quando considerada a sua ligação com o servidor.

2.5.4 Tratamento dos Pacotes de Saída das Respostas HTTP

Pacotes de saída podem surgir de uma transmissão HTTP por duas razões: ou um programa aplicativo passa dados para um dos protocolos de alto nível, o qual, por sua vez, envia uma mensagem para um protocolo de nível inferior e, por fim, gera transmissão em uma rede, ou o programa de protocolo contido no sistema operacional transmite informações. Nos dois casos, um pacote precisa ser enviado por uma determinada interface de rede.

Para isolar a transmissão de pacotes da execução de processos que implementem programas aplicativos, o sistema possui uma fila de saída separada para cada interface de rede.

As filas associadas a dispositivos de saída representam uma parte importante do sistema

operacional. Elas permitem que processos gerem um pacote, o coloquem em uma fila de saída e continuem a execução sem esperar que o pacote seja enviado. Enquanto isso, o hardware pode continuar a transmitir pacotes. Se o hardware estiver inativo quando um pacote chegar, o processo que estiver executando uma saída colocará seu pacote em uma fila e chamará uma rotina para inicializar o hardware. Quando a operação de saída é concluída, o hardware interrompe a CPU. A rotina de processamento de interrupções (específica para o dispositivo), retira da fila o pacote que acaba de ser enviado. Caso haja mais pacotes na fila, a rotina de processamento de interrupções retorna da interrupção, permitindo que o processamento normal continue. Assim, do ponto de vista do processo IP, a transmissão de pacotes corre automaticamente em segundo plano. Enquanto houver pacotes em uma fila, o hardware continuará a transmiti-los. O hardware só precisa ser inicializado quando o IP deposita um pacote em uma fila vazia.

Evidentemente, cada fila de saída possui capacidade limitada, e pode ficar lotada se o sistema gerar pacotes mais rapidamente que o hardware da rede pode transmiti-los. Presume-se que tais casos sejam raros mas, se efetivamente ocorrerem, processos que gerem pacotes precisarão fazer uma escolha: descartar o pacote ou efetuar um bloqueio até que o hardware conclua a transmissão de um pacote e libere mais espaço [14].

2.5.5 Tratamento de Pacotes pelos Processos TCP e IP até a Interface de Rede

Assim como a entrada, a saída do TCP é complexa. Conexões precisam ser estabelecidas, dados precisam ser dispostos em segmentos, e os segmentos precisam ser retransmitidos até que as mensagens de confirmação cheguem. Uma vez colocado em um datagrama, um segmento pode ser passado ao IP para roteamento e entrega. O sistema operacional emprega dois processos TCP para administrar a complexidade. O primeiro, denominado *tcpout*, gerencia a maior parte dos detalhes de segmentação e transmissão de dados. O segundo, denominado *tcptimer*, programa a execução de programação de tempos de retransmissão e avisa ao *tcpout* quando um segmento precisa ser retransmitido [14].

O processo *tcpout* usa uma porta para sincronizar entradas provenientes de vários processos. O TCP se baseia em *streams*, o que significa que ele permite a um programa aplicativo

enviar alguns bytes de dados por vez. Conseqüentemente, os itens encontrados na porta não correspondem a pacotes ou segmentos individuais. Em vez disso, um processo que emite dados os insere em um dispositivo temporário de saída e coloca uma só mensagem na porta, comunicando ao TCP que mais dados foram escritos. O processo *tcptimer* deposita uma mensagem na porta sempre que um temporizador expira e o TCP precisa retransmitir um segmento. Assim, podemos imaginar a porta como uma fila de eventos a serem processados pelo TCP, cada evento pode causar transmissão ou retransmissão de um segmento.

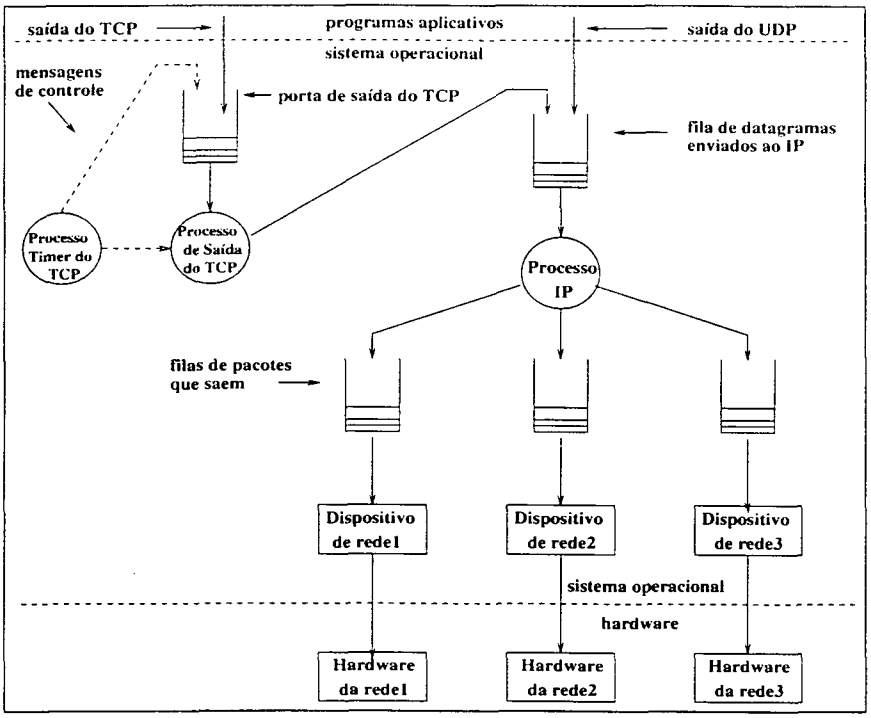


Figura 2.5: Controle de fluxo na saída de dados do TCP.

Depois de produzir um segmento, o TCP passa-o ao IP para a entrega. O IP escolhe uma interface de rede pela qual o datagrama precisa ser enviado, e passa o datagrama ao processo de saída de rede correspondente. A Figura 2.5, retirada de [14], resume o fluxo de informações entre um programa aplicativo e o hardware de rede durante a saída. Um programa aplicativo executado como um processo separado, chama rotinas do sistema para passar a *stream* para o TCP. Na saída TCP, o processo que executa um programa aplicativo chama uma rotina do

sistema para transferir dados pelo sistema operacional e insere-os em um dispositivo temporário. O processo aplicativo informa ao processo de saída TCP que novos dados estão esperando para serem enviados. Ao ser executado, o processo de saída do TCP divide a *stream* em segmentos e encapsula cada segmento em um datagrama IP para entrega. Por fim, o processo de saída do TCP insere o datagrama IP na porta da qual o IP irá extrai-lo e enviá-lo.

2.5.6 Modelos de Filas para Servidores Web

As requisições tratadas em caches utilizam três recursos de um servidor *proxy*: CPU, disco e interface de rede. Para cada requisição, a demanda de serviço pode ser dividida em três partes: tempo de CPU, tempo de leitura e gravação no disco e tempo de processamento na interface de rede.

O tempo de CPU representa o tempo gasto pela CPU para fazer a interpretação da requisição, o empacotamento e o desempacotamento dos segmentos TCP e datagramas IP, o controle da transferência dos dados do disco para a memória e desta para a interface de rede, entre outros aspectos da manipulação de uma requisição HTTP.

O tempo de leitura é o tempo gasto pelo disco para transferir o arquivo solicitado para a memória. O tempo de gravação (requisições POST e PUT) é o tempo gasto para transferir o arquivo da memória para o disco. O tempo de processamento na interface de rede é o tempo gasto para transferir os dados solicitados para a fila da interface de rede.

Os servidores Web e de cache podem ser modelados de diversas formas. A Figura 2.6 apresenta um modelo considerando o lado do cliente, o qual é composto por oito recursos. Este modelo procura representar um servidor Web típico com servidor *proxy* com cache, que pode ser acessado por qualquer cliente na Internet. O recurso 1 é uma fila de espera que representa o conjunto de clientes. O tempo dispendido nessa fila é o tempo de pensar do cliente, que representa o tempo gasto por um cliente entre o momento em que começa a receber um documento até a requisição de um novo documento. O recurso 2 representa a LAN. O roteador é representado pelo recurso 3, o roteador é modelado como um nó de retardo porque sua latência é muito pequena em relação às dos demais recursos. O enlace que conecta o roteador ao ISP (*Internet Service Provider*) é um canal *full-duplex*. Isso significa que ele pode receber e transmitir

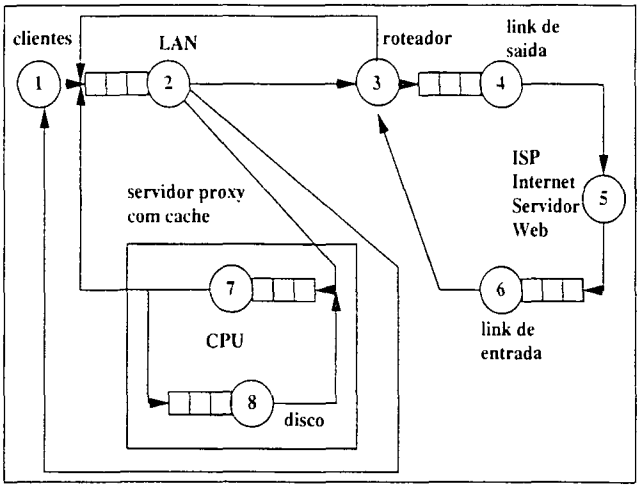


Figura 2.6: Modelo de rede de filas para um servidor *proxy* com cache.

ao mesmo tempo na largura de banda. Assim, se faz necessário modelar os enlaces de saída e entrada separadamente, como dois recursos independentes (recursos 4 e 6, respectivamente). Os atrasos no ISP, seu enlace com a Internet, a própria Internet e os servidores remotos são representados pelo recurso 5. Finalizando, um servidor *proxy* com cache é modelado com uma CPU (recurso 7) e um disco (recurso 8) [31].

Em um ambiente com um único servidor Web no *site*, o servidor fica conectado a uma LAN, que é conectada a um roteador, que conecta o *site* ao ISP e depois a Internet. Os documentos que podem ser atendidos pelo servidor Web ficam armazenados na mesma máquina onde o servidor Web executa. Este processo também pode ser representado por um modelo de rede de filas. A Figura 2.7 apresenta este modelo, o qual leva em conta apenas o lado do servidor Web. Este modelo procura representar um servidor Web típico, que pode ser acessado por qualquer cliente na Internet. Os enlaces de entrada (recurso 1), de saída (recurso 6) e a LAN (recurso 3) são modelados por filas independentes de carga. O roteador (recurso 2) é modelado como um nó de retardo e o servidor Web é representado por dois recursos independentes de carga: um para a CPU (recurso 4) e outro para o disco (recurso 5). Os modelos de desempenho para o lado do servidor podem ser usados para responder perguntas como [31]:

- Como avaliar novas estratégias para a operação do servidor, incluindo a replicação de

documentos populares em discos separados?

- Qual é o impacto do uso de *scripts* CGI?
- Qual é a taxa máxima de processamento da página Web?
- Qual é o impacto do uso da compactação sobre grandes objetos multimídia?

Essa é uma lista parcial das avaliações feitas por administradores de servidor Web que trabalham com os modelos de desempenho.

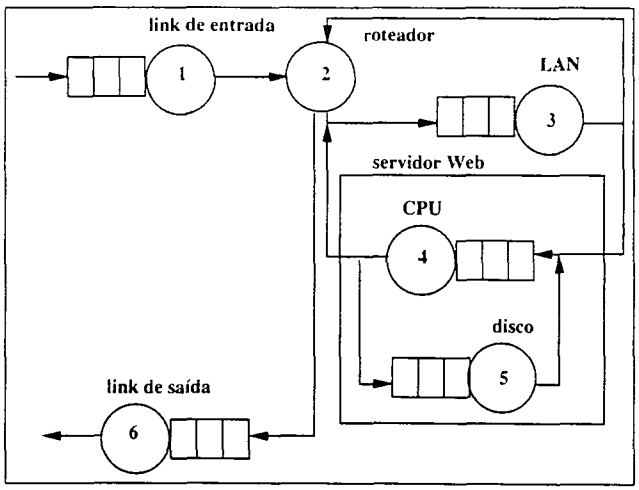


Figura 2.7: Modelo de rede de filas para um servidor Web.

2.6 Trabalhos Relacionados

O crescimento da Web motivou várias pesquisas que buscam caracterizar o tráfego Web, com o objetivo de propor novas soluções para melhorar o seu desempenho e a sua escalabilidade. A análise criteriosa e o entendimento das características de carga da Web são fundamentais para um bom planejamento e implementação de servidores e serviços Web. Nesta seção são apresentados os principais trabalhos correlatos ao tema desta dissertação.

2.6.1 Estudos dos Fatores que Influenciam o Tempo de Serviço

Como vimos na seção anterior, vários fatores podem influenciar o tempo de serviço de uma requisição HTTP. Entre alguns dos fatores estão o protocolo, o desempenho do servidor, o sistema operacional, a plataforma de hardware, a largura da banda de rede e a carga de trabalho [12, 27]. Alguns destes fatores tem profundo impacto no desempenho. Por exemplo, o número de usuários cresce a cada dia e, conseqüentemente, o número de conexões também cresce. Com o crescimento do número de conexões o volume de tráfego aumenta e, com isso, a perda de pacotes. A perda de pacotes é um dos maiores problemas para a transferência de arquivos [10]. Os diferentes navegadores e servidores com diferentes versões do HTTP também têm grande influência no desempenho da Web. As conexões que atravessam redes de longa distância experimentam maiores atrasos. Foi observada queda significativa no desempenho do servidor quando o RTT (*Round Trip Time*) aumenta [7]. Com um RTT de 200 milissegundos, o desempenho de um servidor Web foi reduzido em 54% quando comparado com uma simulação sem RTT. Foram feitos testes com cargas de trabalho em ambientes que simulam uma Intranet, praticamente sem latência de rede, e a Internet com suas variações de latência [20]. Estes testes mostraram que o problema causado por grandes latências é devido ao fato de que o servidor precisa manter um processo ocupado com a requisição até que o último pacote de dados seja enviado ao usuário.

As conexões persistentes permitem que várias mensagens sejam trocadas através de apenas uma conexão TCP, diminuindo os tempos de serviço. Avaliações realizadas mostram uma diminuição dos tempos de serviço obtidos [28]. Em uma das simulações, foi verificada que a diminuição dos tempos de serviço, com o uso de conexões persistentes, é significativa, variando entre 23% e 50%.

2.6.2 Modelos para Tamanho de Resposta e Tempo de Serviço

Estudos revelam grande variabilidade dos tamanhos dos objetos que trafegam na Internet. O tamanho dos objetos é um dos parâmetros mais medidos e avaliados na literatura relacionada ao tráfego HTTP. A maior parte das medições indica que os valores desse parâmetro são melhor descritos por uma distribuição de cauda pesada. *Pareto* é a distribuição preferida para modelar os tamanhos dos objetos transferidos. Uma variável aleatória x segue distribuição de cauda

pesada se

$$\overline{F}(x) = P[X > x] \sim x^{\alpha}, \text{ para } x \rightarrow \infty, 0 < \alpha < 2,$$

onde $\overline{F}$ é o complemento de F que, por sua vez, é a função de distribuição cumulativa (CDF). Variáveis que seguem essa distribuição têm variância infinita, e se $\alpha < 1$, têm média infinita. A probabilidade de valores grandes para x têm valor não desprezível. A função de densidade de probabilidade (PDF) da distribuição de *Pareto* é

$$f(x) = \alpha k^{\alpha} x^{-(\alpha-1)},$$

e a função de distribuição cumulativa é

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

Há outros trabalhos que utilizam a distribuição *Lognormal* e a distribuição *Weibull* [1, 4, 5, 9, 15]. A função de distribuição cumulativa da distribuição *Lognormal* é

$$F(x) = \frac{1}{x\sigma\sqrt{2\pi}} e^{-(lnx-\mu)^2/2\sigma^2}, \; x \geq 0; \; \sigma > 0,$$

onde μ é a média do logaritmo natural dos elementos e σ é o desvio padrão do logaritmo natural dos elementos. Sua função de densidade de probabilidade da distribuição é

$$f(x) = \Phi(\frac{ln(x)}{\sigma}), x \geq 0; \; \sigma > 0,$$

onde Φ é a função de distribuição cumulativa da distribuição *Normal*.

A função da distribuição cumulativa da distribuição *Weibull* é

$$F(x) = 1 - e^{-(\frac{x}{\alpha})^{\beta}},$$

e a sua função de densidade de probabilidade da distribuição é

$$f(x) = \frac{\beta}{x}(\frac{x}{\alpha})^{\beta} e^{-(\frac{x}{\beta})^{\alpha}},$$

onde α é o parâmetro de escala e β é o parâmetro de forma. Os dois parâmetros devem ter valores maiores do que zero.

Para modelar os tamanhos das respostas, alguns autores propõem modelos diferentes para a cauda e para o corpo da distribuição. Em um dos trabalhos que faz utilização de mais de uma distribuição como modelo [9], a cauda da distribuição dos tamanhos foi modelada utilizando a distribuição de *Pareto* (parâmetros α = 1,1 e k = 133Kbytes) e o corpo é modelado utilizando distribuição *Lognormal* (parâmetros μ = 9,357 e σ = 1,318). Alguns programas são usados para facilitar a obtenção dos parâmetros das distribuições usadas como modelo. O programa *aest* (*alpha estimator*) [17] é usado para obter o parâmetro α da distribuição de *Pareto* que modela a cauda. Em um dos casos de sua utilização [4], os tamanhos são modelados e os resultados apresentados são similares ao do trabalho descrito acima [9]. O programa estimou um índice α = 1,37 quando os arquivos tratados tinham tamanhos entre 1MB a 4MB.

Alguns trabalhos apresentam modelos de cargas de trabalho de diferentes locais. Em um dos exemplos [1], são apresentadas modelagens de várias cargas, são utilizados como modelos as distribuições de *Weibull* (carga DEC1 com parâmetros α = 3,240 e β = 0,48 e carga DEC2 com parâmetros α = 3,822 e β = 0,48) e *Lognormal* (carga *Korea* com parâmetros μ = 7,54 e σ = 1,78 e carga AOL com parâmetros μ = 7,64 e σ = 1,49).

Não é do conhecimento dos autores a existência de modelos publicados para os tempos de serviço. Mesmo com o grande volume de registros de acessos disponíveis, existem poucos trabalhos de caracterização de tempos de serviço de caches publicados, dificultando a obtenção de dados estatísticos e também o entendimento das razões de longos tempos de resposta observados freqüentemente no atendimento de requisições.

A caracterização dos tempos de serviço em servidores *proxy-cache* é abordada em dois trabalhos publicados na literatura. O primeiro trabalho [34] apresenta estudos com três caches de diferentes países, com o objetivo de apresentar uma análise dos tempos de resposta em servidores *proxy-cache* e as diferenças nas taxas de serviço obtidas nos diferentes países. Na caracterização dos tempos de serviço nos servidores *proxy-cache* foram utilizados três parâmetros: o tempo, que representa o tempo da transmissão; o tamanho, que representa o número de bytes destinados ao cliente; e o *throughput*, que representa a taxa efetiva de transmissão. Os resultados mostram

que o tempo médio de serviço para os *hits* variou entre 0,55 e 3.5 segundos para as três cargas e o tamanho médio das respostas variou entre 8 e 12 KBytes. No caso dos *misses*, o tempo médio de resposta variou entre 5 e 13 segundos, enquanto o tamanho médio variou entre 10 e 17 KBytes. As médias obtidas para os *hits* foram sempre menores dos que as obtidas para os *misses*.

Um estudo sobre a correlação entre o tamanho e o tempo de serviço foi apresentado no mesmo artigo [34]. Os autores apresentaram gráficos nos quais esta correlação pôde ser verificada visualmente para objetos com tamanhos maiores do que 32 KBytes. Os autores não apresentaram valores para a correlação entre o tamanho da resposta e o tempo de serviço, nem relações matemáticas entre as grandezas envolvidas.

Os resultados relacionados ao *throughput* mostraram que as taxas obtidas nos conjuntos de *hits* são bem superiores às obtidas com os *misses*. As taxas das transmissões que são *hits* no cache variaram entre 9 e 264 Kbits/s e as taxas dos *misses* sempre foram menores do que 1 Kbit/s. A grande variabilidade obtida nas taxas pode ser explicada por fatores como a grande variabilidade nos tamanhos dos arquivos e na largura de banda dos enlaces com o servidor *proxy-cache*.

O segundo trabalho sobre caracterização dos tempos de resposta em servidores *proxy-cache* encontrado na literatura [42] mostra que a correlação entre o tamanho e o tempo cresce de acordo com o crescimento do tamanho dos arquivos processados. Para este trabalho os autores analisaram pouco mais de um milhão de registros cujas respostas são exclusivamente *hits*. Os registros foram divididos em três faixas de tamanho, sendo considerados como MICE os arquivos menores que 30 Kbytes, ELEPHANTS os arquivos maiores ou iguais a 30 Kbytes e menores ou iguais a 500 Kbytes, e MAMMOTHS os arquivos com mais de 500 Kbytes. As correlações entre os tamanhos e os tempos de resposta foram calculadas nestes grupos. O grupo MICE apresentou a menor correlação (0,016) enquanto o grupo MAMMOTHS apresentou a maior correlação (0,538). Os resultados mostram que o tempo de serviço varia muito, independentemente do tamanho do arquivo. Conclui-se que, devido à esta grande variação, não é possível predizer os tempos de resposta levando em consideração apenas o tamanho dos arquivos. Os autores mostram que a variabilidade dos tempos de resposta para arquivos de mesmo tamanho compromete a precisão de qualquer modelo de correlação, seja este linear ou não.

2.6.3 Caracterização da Carga

Um dos principais objetivos da caracterização da carga de um sistema é a identificação de características que se mantém constantes em várias instâncias da carga. Estas características são denominadas invariantes. Na literatura são encontrados estudos que apontam características invariantes nas cargas dos servidores Web [5] e nas cargas de servidores *proxy* [2].

Em um dos trabalhos [2], para apontar os invariantes das cargas dos servidores *proxy*, os autores analisaram dez diferentes *logs* de acesso, que foram coletados em ambientes diversos tais como universidades e instituições de pesquisa. Os conjuntos de dados foram coletados em intervalos de tempo desde uns poucos minutos até um ano de atividade. Foram identificadas dez características no tráfego dos servidores *proxy* que se aplicam a todas as cargas analisadas. Estas características representam, potencialmente, constantes universais a todos os servidores *proxy*, denominadas invariantes. As invariantes encontradas são resumidas na Tabela 2.3, conforme [2].

2.7 Considerações finais

Tendo em vista estes trabalhos, esta dissertação tem dois objetivos. O primeiro é dar continuidade à caracterização de carga dos servidores *proxy-cache*, apresentando caracterizações de registros de carga recente, bem como discutindo a evolução da carga. Para isso, os resultados obtidos na caracterização feita nesta dissertação são comparados com os resultados de caracterizações descritas na literatura. O segundo objetivo é apresentar um modelo para os tempos de resposta às requisições, comparando-o com modelos para os tamanhos das respostas.

Nesta dissertação são analisadas cargas obtidas no mês de outubro de 2002, totalizando mais de dezoito milhões de requisições, o que corresponde a um fator de pelo menos dez vezes o número de requisições analisadas em cada um dos trabalhos citados anteriormente.

A carga analisada é descrita no próximo capítulo.

Constante	Nome	Descrição
1	Mediana dos tamanhos	Aproximadamente 2 KBytes
2	Média dos tamanhos	Menor que 27 KBytes
3	Tipos dos objetos acessados	90 a 98% dos objetos acessados são para documentos HTML, arquivos de imagem e programas CGI
4	Tipos dos arquivos em bytes	Dominância dos bytes transferidos pelas imagens
5	Acesso a um único servidor	Menos de 11% do acesso é feito a um único servidor
6	Servidores referenciados uma única vez	Menos de 5% dos servidores são referenciados uma única vez
7	Concentração de acessos aos servidores	25% dos servidores são responsáveis por 80 a 95% das referências
8	Concentração de acessos aos bytes	25% dos servidores são responsáveis por 90% dos bytes acessados
9	Taxa de sucesso	88 a 99% das referências feitas ao *proxy* resultam em sucesso de transferência para os clientes
10	Auto-similaridade	$0{,}59 \leq H \leq 0{,}94$

Tabela 2.3: Invariantes para servidores *proxy*.

Capítulo 3

Descrição da Carga Analisada

Neste capítulo são apresentadas as cargas trabalhadas e os resultados obtidos na análise dos registros de acesso de quatro servidores. A finalidade deste estudo é avaliar a evolução da carga nos últimos anos, comparando os resultados de caracterização obtidos com resultados já publicados na literatura [1, 2, 5, 9, 15]. Sempre que for pertinente, uma comparação dos resultados é apresentada.

Na seção 3.1 mostra a descrição da carga coletada. Na seção 3.2 descrevemos o formato dos arquivos de registro e quais os campos foram utilizados neste trabalho. Na seção 3.3 é apresentada uma análise dos dados processados. Este processamento inicial descreve a carga de trabalho, pois ela influencia diretamente o desempenho de caches na Web. Na seção 3.4 é feita uma avaliação da evolução da carga de trabalho em caches Web e, finalizando este capítulo, são apresentadas as considerações finais do capítulo na seção 3.5.

3.1 Descrição da Carga Coletada

Para a obtenção dos resultados apresentados neste trabalho foram utilizados registros obtidos do IRCache, que é um projeto de cache Web do laboratório NLANR [37] originalmente criado pela *National Science Foundation*. A hierarquia de servidores cache do NLANR registra os acessos de toda a Internet. Os servidores do IRCache distribuem-se em domínios geograficamente localizados em pontos estratégicos para prover melhor distribuição de carga, conforme mostra a

Figura 3.1. Os caches utilizados localizam-se da seguinte forma:

Cache PB - pb.us.ircache.net - Pittsburgh, Pensilvânia

Cache SV - sv.us.ircache.net - Silicon Valley, Califórnia

Cache SD - sd.us.ircache.net - San Diego, Califórnia

Cache RTP - rtp.us.ircache.net - Research Triangle Park, Califórnia

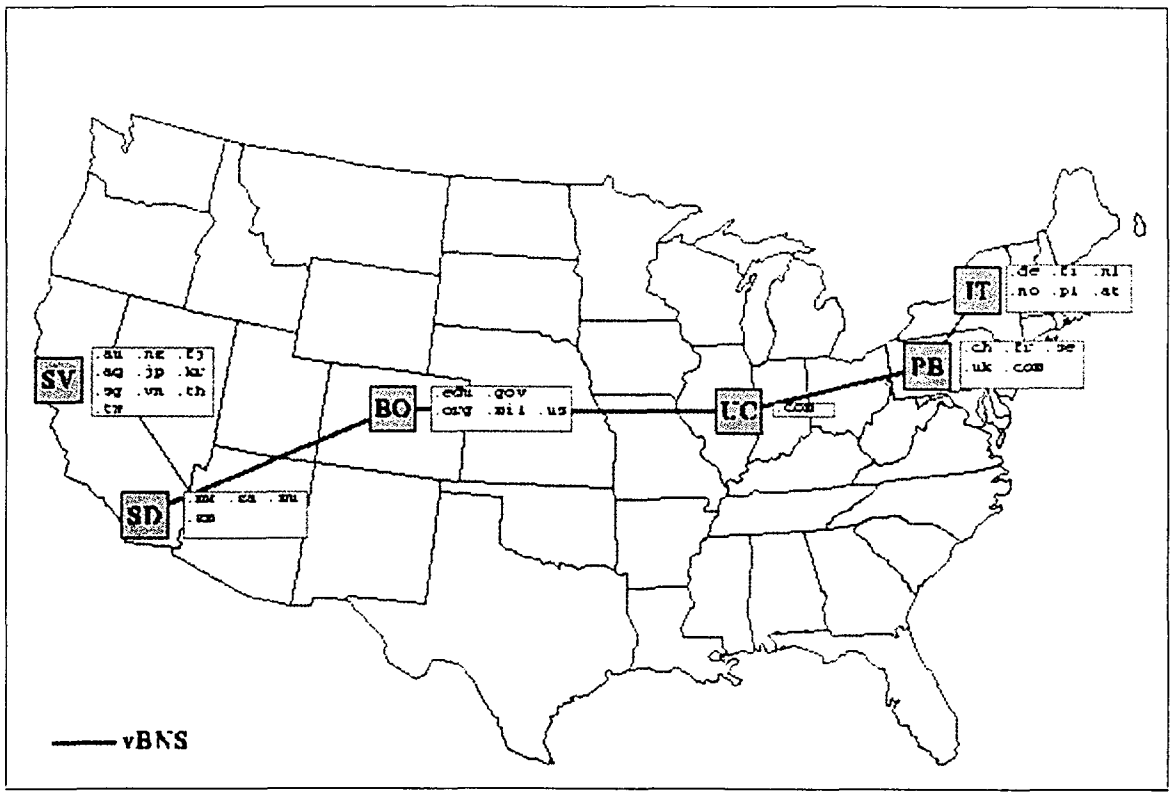


Figura 3.1: Hierarquia de Caches NLANR.

A Figura 3.1 [48] mostra o VBNS (*Very high performance Backbone Network Service*) com as conexões de rede entre os sistemas de cache e alguns domínios que são tratados em cada cache. Os caches PB e SD fazem parte do *backbone* de alta velocidade VBNS. Estes caches são responsáveis pelo encaminhamento das requisições aos domínios `.com`, `.net`, `.org`, `.edu`, `.gov`, `.mil` e `.us`, para os outros caches. SD, na costa oeste, conecta-se a caches da África do Sul, México,

Brasil e Rússia. SV, também na costa oeste, não se conecta a outros caches do NLANR, com exceção do cache BO (que atende requisições aos domínios `.th`), e tem pontos de conexão com caches do Japão, Austrália, e outros países do sudeste asiático. PB, na costa leste, interliga-se a caches do Reino Unido, Suécia, Holanda e recebe requisições de caches de outros *backbones* europeus. RTP, não mostrado no mapa, atende requisições dos domínios `.com`, `.net` e `.org`. Uma listagem completa dos domínios tratados pelos caches pode ser obtida em [37].

Os servidores cache cuja carga foi analisada estão localizados na Internet 2 americana. Cada servidor trata milhares de requisições diariamente, oriundas de usuários cuja distribuição geográfica contempla todos os continentes. Assim, podemos considerar que estas cargas refletem o comportamento de toda a Internet, pois não dependem de um conjunto específico de clientes ou de servidores.

Os registros de acesso analisados foram coletados no período de 28/10/2002 a 31/10/2002.

3.2 Formato dos Arquivos de Registro

Cada arquivo de registro contém registros de um dia, sendo que, é disponibilizado um registro por linha. Uma linha do arquivo tem o seguinte formato:

```
1              2   3              4            5    6   7                                  8 9       10
1035763216.182 124 41.42.156.221 TCP_HIT/200 928 GET http://www.azlyrics.com/lup.gif - NONE/- image/gif
```

Cada linha do arquivo representa uma requisição ao cache e contém dez campos, separados por um espaço, que são descritos a seguir:

1. Encerramento da conexão: tempo em que o pedido foi completado no formato do *Unix time* (segundos desde 01 de janeiro de 1970), com arredondamento de milissegundos.

2. Tempo de serviço: tempo de duração da conexão, em milissegundos; este tempo é contado desde o momento em que a conexão é aceita até o momento em que o último pacote de transferência é copiado na interface de rede; para conexões HTTP persistentes, este tempo é entre a leitura do primeiro byte da requisição e a escrita do último byte da resposta.

3. Endereço do cliente: endereço IP do cliente transformado por uma máscara: o endereço do cliente permanece o mesmo para todas as requisições em um arquivo de registro, mas não é o mesmo entre arquivos de registro.

4. Código de resposta HTTP e código de resultado do *Squid*: descrevem como a requisição foi tratada localmente (acertos, erros, dentre outros); os códigos são descritos no Apêndice A.

5. Tamanho da resposta: número de bytes retornados ao cliente.

6. Método da requisição: o método de requisição HTTP; os métodos são descritos no Apêndice A.

7. URL: a URL requisitada; argumentos de consulta CGI não são registrados.

8. Identificação do usuário: não identificado nos registros do IRCache, substituído por "-".

9. Hierarquia e endereço do servidor: descrição de onde e como a requisição e os objetos foram encontrados.

10. Tipo do arquivo: o tipo do arquivo servido pelo cache (imagem, texto, áudio, vídeo, aplicação, entre outros).

Neste trabalho são explorados os campos 2, 4, 5, 6 e 10. O campo 2 é usado como sendo o tempo de serviço, o campo 5 como o tamanho da resposta, os demais campos são usados para a geração de dados atuais da carga e na comparação com resultados anteriormente publicados.

3.3 Análise da Carga

Esta seção apresenta os resultados obtidos nos processamentos iniciais, com o objetivo de descrever as características gerais da carga. Os resultados são comparados com os resultados apresentados na literatura.

Após a obtenção dos registros foram utilizadas rotinas para validá-los, pois 28.511 registros encontravam-se com tamanho zero. Estes registros foram eliminados por serem, possivelmente, falhas de registro. Foram mantidos 18.630.327 registros para obtenção das primeiras informações. A Tabela 3.1 apresenta o total de requisições por cache e o número médio de requisições analisadas por dia para cada cache, sendo que, para dois caches foram considerados três dias e para

os outros dois quatro dias. Nesta tabela também são apresentados o total de bytes transferidos e a média em Mbytes trasferidos por dia. Na seqüência são apresentados os tamanhos médios dos arquivos, estes variaram entre 10,19 e 13,98 Kbytes. Posteriormente são apresentados os valores obtidos para a mediana que variou entre 0,7 e 1,3 Kbytes. Nas últimas duas linhas da tabela são apresentados os tamanhos dos menores e dos maiores arquivos encontrados. Os menores arquivos variaram entre 12 e 30 bytes e os maiores arquivos encontrados nos registros estão entre 156 e 377 Mbytes. Posteriormente são descritos novos filtros e então os resultados obtidos são comparados com os dos trabalhos anteriores.

Característica	cache SD	cache SV	cache PB	cache RTP
Duração do *log*	3 dias	4 dias	4 dias	3 dias
Data de início	29/10/2002	28/10/2002	28/10/2002	29/10/2002
Data de término	31/10/2002	31/10/2002	31/10/2002	31/10/2002
Total de requisições	2.227.419	3.244.374	6.368.561	6.789.973
Média diária de requisições	742.473	811.093	1.592.140	2.263.324
Total de Mbytes transferidos	30.413	32.278	65.057	77.959
Mbytes por dia	10.137	8.069	16.264	25.986
Tamanho médio em Kbytes	13,98	10,19	10,46	11,76
Mediana em Kbytes	1,0	1,3	0,7	0,9
Menor arquivo em bytes	17	30	12	12
Maior arquivo em Mbytes	311	156	281	377

Tabela 3.1: Descrição da carga analisada por cache.

A Tabela 3.2 mostra o método da requisição referente ao protocolo HTTP. Um método de pedido informa ao servidor a ação que deve ser executada sobre o recurso identificado pelo endereço do pedido. Para detalhes sobre métodos HTTP ver Apêndice A. Nos arquivos coletados, mais de 98% das requisições são GET, ou seja, a finalidade do pedido é ler e transmitir o conteúdo atual do recurso solicitado. Em [4], o percentual de GET ultrapassa 99%. O método POST obteve apenas 1,16%. Já a utilização dos métodos HEAD, PUT, CONNECT e DELETE teve

uso inexpressivo e suas ocorrências foram então agrupadas no item OUTROS. Estes resultados mostram que, na grande maioria das vezes, as requisições são para a exibição de um arquivo, ou seja, pode-se verificar que a Internet é amplamente utilizada para a consulta de informações. As Tabelas 3.2 e 3.3 apresentam dados correspondentes ao agrupamento dos registros dos quatro caches.

Método	Ocorrências	Fração(%)	Mbytes por Método	Fração dos bytes(%)
GET	18.385.886	98,69	205.145	99,73
POST	217.113	1,16	549	0,26
OUTROS	27.328	0,15	13	0,01
Total	18.630.327	100,00	205.707	100,00

Tabela 3.2: Registros por tipo de método.

A Tabela 3.3 apresenta as ocorrências dos códigos retornados pelo protocolo HTTP em resposta à execução das requisições. O código de resposta informa se o pedido processado obteve sucesso ou falhou. As respostas são codificadas em classes. A classe 2xx indica que foi encontrado um objeto válido no servidor e este foi retornado ao cliente. A classe 3xx é usada para informar ao cliente que uma ação adicional é necessária para completar a requisição. As classes com maior ocorrência são a classe 2xx (*Successfull*), com mais de 56% das requisições, e a classe 3xx (*Redirection*), com mais de 33% das requisições. O código 200 é retornado quando a requisição do cliente foi atendida com sucesso, ou seja, em caso de *hit*, o objeto é transferido do cache para o cliente e, em caso de *miss*, o objeto é transferido do servidor para o cache e do cache para o cliente. O código 304 indica que o cliente executou uma solicitação condicional para identificar se o objeto foi modificado, mas o objeto não foi modificado desde a data e hora especificados no campo *If-Modified-Since*. Portanto, o servidor enviou a informação de que o objeto encontrado no cache é válido.

Comparando os dados obtidos com os do trabalho [2] percebe-se menor ocorrência de requisições com o código 200 na carga mais atual (54% contra 80% em média), e um aumento considerável de ocorrências do código 304 (10% contra 27% em média). Isto significa que o número de buscas condicionais teve um crescimento significativo indicando que muitos dos do-

Código HTTP	Ocorrências	Fração(%)	Total em Mbytes	Fração dos bytes(%)
200	10.203.871	54,77	175.029	85,08
206	287.891	1,55	25.197	12,25
302	1.045.968	5,61	600	0,29
304	5.128.740	27,53	1.358	0,66
404	339.707	1,82	1.085	0,53
504	969.598	5,20	1.208	0,59
Outros	683.063	3,52	1.230	0,60
Total	18.630.327	100,00	205.707	100,00

Tabela 3.3: Registros por *status code* HTTP.

cumentos requisitados já estavam no cache.

Após a obtenção das informações gerais de registros de acesso foram executados programas para filtrar dados que não seriam utilizados nesta pesquisa. Foram selecionados apenas os registros que tratavam de requisições TCP, cujo método de requisição era GET e que o código de resposta do HTTP era *successfull* 2xx (código de resposta da classe de sucesso) ou 304 (código de resposta condicional), contabilizando 15.400.490 registros, representando 82,53% do total de registros coletados.

Na Tabela 3.4 são apresentados os dados referentes aos registros que foram mantidos e que serão utilizados nos processamentos realizados no complemento deste trabalho. Nesta tabela são apresentados o total de requisições mantidas para cada cache, a média de requisições diárias, o total de Mbytes transferidos e a média em Mbytes trasferidos por dia. Na seqüência são apresentados o tamanho médio e a mediana dos arquivos. O tamanho médio dos arquivos variou entre 12,16 e 19,68 Kbytes e a mediana variou entre 0,7 e 1,5 Kbytes. Os menores arquivos têm tamanhos entre 17 e 140 bytes. Os maiores arquivos permanecem com os mesmos tamanhos apresentados na Tabela 3.1, mesmo após a aplicação dos filtros adicionais.

Nas Tabelas 3.5 e 3.6 são apresentadas as informações referentes as requisições 2xx e 304 respectivamente. Os registros das requisições 2xx foram utilizados para obter os dados estatísticos para avaliação dos tipos de arquivos transmitidos e os registros das requisições 304 foram utilizados na geração dos modelos que serão descrito nos capítulos posteriores.

Característica	cache SD	cache SV	cache PB	cache RTP
Total de requisições	1.534.158	2.649.572	5.178.754	6.038.006
Média diária de requisições	511.386	662.393	1.294.688	2.012.668
Total de Mbytes transferidos	29.481	31.461	63.366	76.734
Mbytes por dia	9.827	7.865	15.841	25.578
Tamanho médio em Kbytes	19,68	12,16	12,53	13.01
Mediana em Kbytes	0,7	1,5	0,7	0.9
Menor arquivo em bytes	132	140	17	17
Maior arquivo em Mbytes	311	156	281	377

Tabela 3.4: Descrição da carga analisada após a filtragem aplicada.

Na Tabela 3.5 são apresentados o número de registros das requisições 2xx que são responsáveis por 10.273.017 registros. Também são apresentados a média de requisições diárias, o total de Mbytes transferidos e a média em Mbytes trasferidos por dia. Na seqüência são apresentados o tamanho médio e a mediana dos arquivos. O tamanho médio dos arquivos variou entre 17,28 e 27,75 Kbytes e a mediana variou entre 2,21 e 3,41 Kbytes. Os menores arquivos têm tamanhos entre 106 e 162 bytes. Os maiores arquivos encontrados nos registros estão entre 156 e 377 Mbytes.

Comparando com trabalhos anteriores [1, 30] é possível verificar que a mediana teve pequena redução, o tamanho médio dos arquivos teve aumento considerável, e a média de requisições diárias e o total de bytes transferidos têm aumentado. O maior arquivo encontrado em cada registro têm tamanho bem superior aos analisados anteriormente, dado que os maiores arquivos tinham tamanho entre 6,7 e 46 Mbytes [1]. Nos registros processados para obtenção dos dados deste trabalho, os maiores arquivos estão entre 156 e 377 Mbytes mostrando significativo aumento nos maiores arquivos transmitidos.

Na Tabela 3.6 são apresentados o número de registros das requisições 304 que são responsáveis por 5.127.473 registros. Também são apresentados a média de requisições diárias, o total de Mbytes transferidos e a média em Mbytes trasferidos por dia. Na seqüência são apresentados

Característica	cache SD	cache SV	cache PB	cache RTP
Total de requisições	1.083.415	1.851.472	3.322.450	4.015.680
Média diária de requisições	361.138	462.868	830.607	1.338.560
Total de Mbytes transferidos	29.360	31.246	62.883	76.196
Mbytes por dia	9.786	7.811	15.720	25.398
Tamanho médio em Kbytes	27,75	17,28	19,38	19,43
Mediana em Kbytes	2.59	3,41	2,21	2,97
Menor arquivo em bytes	132	162	106	158
Maior arquivo em Mbytes	311	156	281	377

Tabela 3.5: Descrição da carga das requisições 2xx.

o tamanho médio e a mediana dos arquivos. O tamanho médio dos arquivos variou entre 0,27 e 0,28 Kbytes e a mediana ficou em 0,26 Kbytes para todos os caches. Os menores arquivos têm tamanhos entre 17 e 140 bytes. Os maiores arquivos encontrados nos registros estão entre 1 e 3 Kbytes. Estes registros apresentam pequena variação dos tamanhos, pois são transmitidas apenas informações de que os dados disponibilizados nos cache ainda são válidos.

A Tabela 3.7 apresenta os dados referentes aos tipos de arquivos encontrados nos caches. Para obtenção dos dados apresentados nesta tabela foram utilizados apenas os registros da classe 2xx sendo que, os registros dos quatro cache foram agrupados. São mostrados os tipos, o número de ocorrências por tipo, a porcentagem de ocorrências por tipo, o tamanho médio das requisições em bytes e a fração dos bytes por tipo de arquivo. Observa-se que os tipos mais freqüentes têm tamanho médio pequeno, mas a porcentagem de arquivos com tamanho médio maior que 69 Kbytes chega aos 11%. No item "outros" estão relacionados todos os tipos com freqüência e fração de bytes menor do que 1%.

As requisições de imagens representam uma fração significativa, em torno de 59% das requisições e 21% da fração de bytes. O tamanho médio da imagens ficou em torno de 7 Kbytes. Comparando com trabalhos anteriores é possível observar que a fração das requisições, o tamanho médio e a fração dos bytes correspondentes às imagens têm diminuido com o passar dos

Característica	cache SD	cache SV	cache PB	cache RTP
Total de requisições	450.743	798.100	1.856.304	2.022.326
Média diária de requisições	150.247	199.525	464.076	674.108
Total de Mbytes transferidos	121	214	482	538
Mbytes por dia	40	53	120	179
Tamanho médio em Kbytes	0,28	0,28	0,27	0,27
Mediana em Kbytes	0,26	0,26	0,26	0,26
Menor arquivo em bytes	140	140	17	17
Maior arquivo em Kbytes	3	1	1	1

Tabela 3.6: Descrição da carga das requisições 304.

anos. Isto ocorreu porque a forma de construir as páginas evoluiu de poucas imagens grandes para páginas mais complexas, compostas de um número maior de imagens pequenas. Nos processamentos realizados em trabalhos anteriores, o tamanho médio das imagens era de 8 Kbytes, a fração das requisições era de 68% e a fração dos bytes era de 40% [30]. Os tipos de imagens mais freqüentes são *image/gif* com 35,52% das requisições e *image/jpeg* com 23,39% das requisições.

Os arquivos do tipo texto continuam tendo um grande número de requisições, em torno de 28%. Mesmo com a popularização de programas do tipo *script*, que executam no servidor para gerar dinamicamente as páginas, a fração de arquivos texto se mantém praticamente inalterada. O tipo de arquivo de texto mais popular é o *text/html* com 25,52% das requisições.

As aplicações são responsáveis pela transmissão da maior fração de bytes, neste tipo são agrupados os programas executáveis, arquivos para aplicativos e também arquivos de dados compactados. Os tipos mais freqüentes são o *application/x-javascript* com 5,90% das requisições e *application/octet-stream* com 3,02% das requisições.

Os tipos com maiores tamanhos médios são os tipos de áudio e de vídeo. Estes tipos têm representação em requisições pequenas, mas são responsáveis por mais de 15% da fração de bytes. O tipo mais comum de vídeo é o *video/mpeg* com 0,20% das requisições e de áudio é o tipo *audio/x-pn-realaudio* com 0,08% das requisições.

Tipo	Ocorrências	Fração (%)	Tamanho Médio	Fração dos bytes(%)
imagem	6.093.131	59,31	7.511	21,86
texto	2.929.857	28,52	17.078	23,90
aplicação	1.105.983	10,77	69.119	36,51
-	52.827	0,51	44.361	1,12
vídeo	33.736	0,33	751.069	12,10
áudio	25.744	0,25	316.392	3,89
outros	31.739	0,31	41.338	0,62
Total	10.273.017	100.00	20.382	100,00

Tabela 3.7: Registros por tipo de arquivo. O tamanho é dado em bytes.

As requisições identificadas com "-" referem-se a requisições dinâmicas. A taxa de requisições dinâmicas tem diminuido, mesmo com a popularização de programas que executam nos servidores (*script*), tais como PHP (*Hypertext Preprocessor*), JSP (*Java-Server Pages*) e ASP (*Active Server Pages*). Em trabalhos anteriores a fração das requisições dinâmicas estava em torno de 1% [4, 30]. O crescimento do comércio eletrônico pode contribuir para um possível aumento desta representação, pois geralmente este tipo de serviço faz uso dessas tecnologias.

3.4 Avaliação da Evolução da Carga em Oito Anos

Nesta seção são apresentadas tabelas e gráficos comparativos dos dados obtidos neste trabalho com os já descritos na literatura [1, 4, 30, 33, 36]. São avaliadas as informações referentes a evolução da variabilidade nos tamanhos das requisições e dos acessos por tipo de arquivo.

Na Tabela 3.8 são apresentadas informações referentes à variação dos tamanhos dos arquivos transmitidos. Nesta tabela são mostrados o ano em que os dados foram coletados, o trabalho no qual os dados foram publicados, o cache que fez os registros, o período de referência dos registros em dias, o número de registros avaliados, o tamanho do maior arquivo encontrado no conjunto de registros, expresso em Kbytes. Todos demais tamanhos estão expressos em bytes. Nas colunas subseqüentes são apresentadas a média, a mediana, o desvio padrão e o coeficiente de variação dos tamanhos dos registros. As quatro últimas linhas da tabela, referentes ao ano 2002, referem-se aos dados obtidos neste trabalho. Para facilitar a análise dos dados da Tabela 3.8, criamos

gráficos que facilitam a visualização das variações ocorridas nos aspectos avaliados.

O gráfico da Figura 3.2 apresenta os maiores arquivos encontrados em cada uma das cargas analisadas. Cada ponto deste gráfico representa uma informação da coluna "Maior" da Tabela 3.8. Como já descrito anteriormente, é possível verificar que os maiores arquivos dos caches cresceram em bytes com o passar dos anos, sendo que, a partir de 1999, a tendência de crescimento dos tamanhos dos maiores arquivos aumentou. Nos dados mais recentes os maiores documentos são da ordem 10^8 em bytes, sete ordens de grandeza maiores do que os menores documentos. Estas informações evidenciam o crescimento da variabilidade existente nos tamanhos das requisições.

No gráfico da Figura 3.3 são exibidos os dados das médias dos tamanhos dos arquivos em cada carga. Cada ponto representa uma informação da coluna "Média" da Tabela 3.8. Analisando os pontos percebe-se que as médias dos tamanhos dos arquivos dos caches foram diminuindo com o passar dos anos, com pequena tendência de crescimento nos últimos anos. Os dados mais recentes mostram que os tamanhos médios chegam a 27 Kbytes. Como o tamanho do maior arquivo cresceu muito, indicando que arquivos maiores são transferidos na rede, e a média cresceu pouco, é possível inferir que o acesso a arquivos pequenos têm crescido com o passar dos anos.

Outro aspecto a ser observado é a diminuição da mediana. No gráfico da Figura 3.4 são apresentadas as medianas dos tamanhos dos arquivos encontrados em cada uma das cargas. Cada ponto representa uma informação da coluna "Mediana" da Tabela 3.8. Ao contrário da média, a mediana foi diminuindo um pouco com o passar dos anos. Nas informações das cargas mais recentes a mediana chegou a 2,21 Kbytes mostrando que o acesso a pequenos arquivos cresceu muito. Como o tamanho do maior arquivo aumentou muito, a média cresceu e a mediana diminuiu, isto implica em um aumento na variabilidade nos tamanhos das requisições.

No gráfico da Figura 3.5 são mostrados os desvios padrão encontrados em cada uma das cargas. Cada ponto representa uma informação da coluna "Desvio Padrão" da Tabela 3.8. Como pode ser observado no gráfico, o desvio padrão tem tendência de crescimento, confirmando então o aumento na variabilidade nos tamanhos das requisições.

No gráfico da Figura 3.6 foram plotados os coeficientes de variação encontrados em cada uma

das cargas. O coeficiente de variação é definido como a razão entre o desvio padrão e a média. Valores acima de 1 indicam grande variabilidade nos dados [24]. Cada ponto representa uma informação da coluna "COV" da Tabela 3.8. Como pode ser observado no gráfico, o coeficiente de variação tem tendência de crescimento, confirmando então o aumento na variabilidade nos tamanhos das requisições.

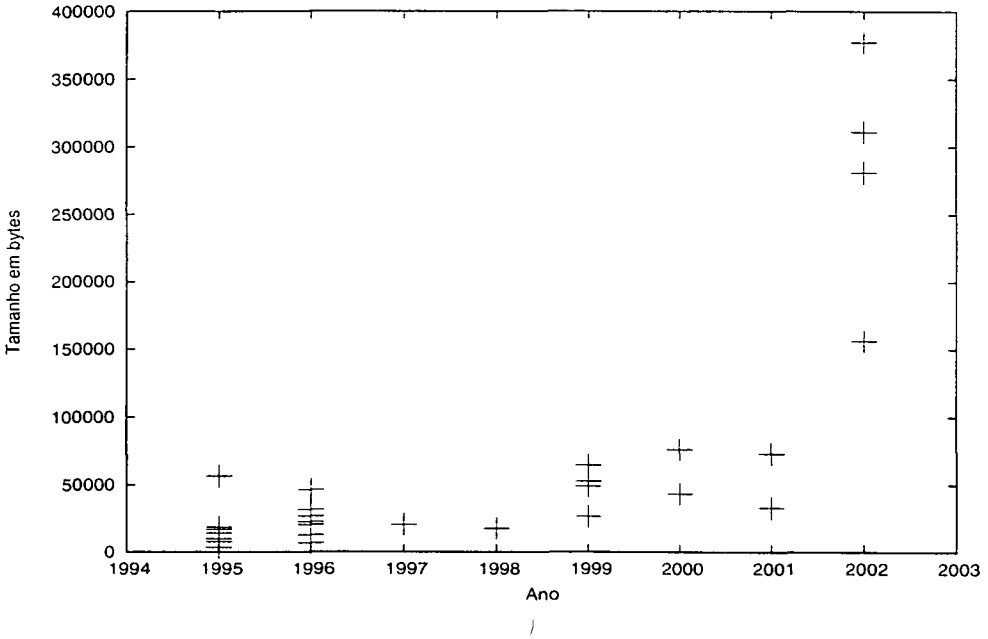


Figura 3.2: Evolução do tamanho do maior arquivo em oito anos.

Na Tabela 3.9 são apresentadas as informações referentes a porcentagem de acessos por tipo de arquivo no período de oito anos. São exibidos o ano em que as informações foram coletadas, o cache e os tipos de arquivos com maior freqüência de requisições. As informações estão dispostas em ordem cronológica para facilitar as observações. A partir dos dados é possível perceber que os tipos HTML e imagens sempre foram os que tiveram a maioria das requisições, apenas nos dados mais recentes as aplicações tiveram maior porcentagem de bytes transferidos. As freqüências de ocorrências têm variado em todos os anos, sendo que a tendência de queda na representação das imagens se mostra constante nos últimos anos. As requisições do tipo HTML tiveram um acréscimo acentuado a partir de 1999, se mantendo com pequena oscilação até os últimos dados apresentados. O tipo CGI, que corresponde as respostas geradas dinamicamente,

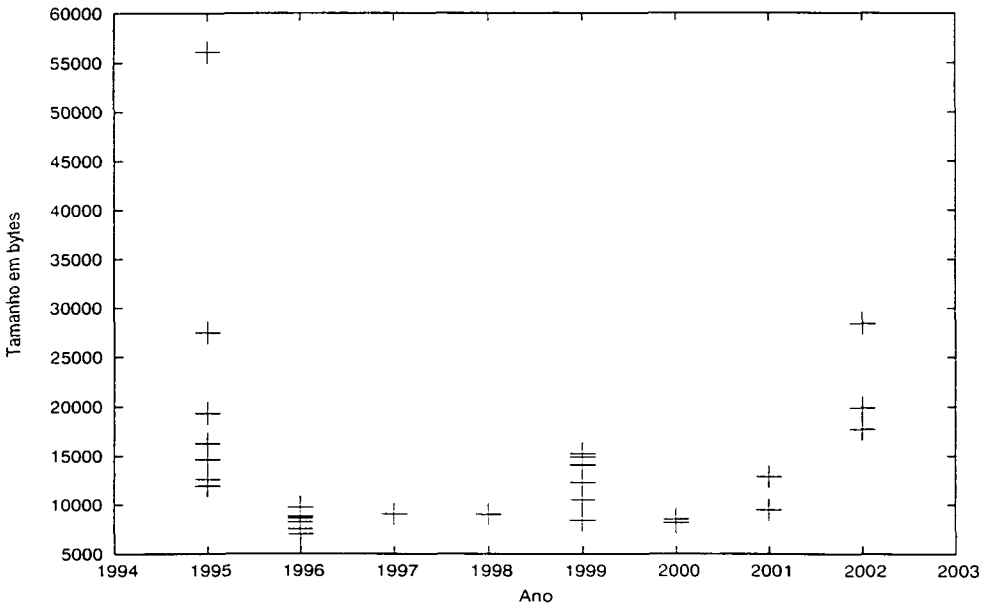


Figura 3.3: Evolução do média dos tamanhos dos arquivos em oito anos.

sofreu grande variação, chegando ter 19,2% da representação das requisições em 1996. A partir de 1998 as requisições geradas dinamicamente tiveram um decréscimo acentuado chegando a ter representação próxima de 1%. Os demais tipos continuam com pequena representação e pequena variação no número de acessos.

Na Tabela 3.10 são apresentadas as informações referentes a porcentagem de bytes por tipo de arquivo no período de oito anos. São exibidos o ano em que as informações foram coletadas, a carga e os tipos de arquivos com maior freqüência de bytes requisitados. A partir dos dados é possível perceber que os arquivos de imagens sempre foram responsáveis pela transmissão da maior porcentagem de bytes. As freqüências de bytes transferidos têm variado todos os anos, sendo que a tendência de queda na representação das imagens se mostra constante nos últimos anos. As requisições do tipo HTML tiveram um acréscimo acentuado na representação da porcentagem de bytes transmitidos a partir de 1999, se mantendo com pequena oscilação até os últimos dados apresentados. O tipo CGI apresenta tendência de queda na porcentagem dos bytes transferidos. Este tipo sofreu grande variação, chegando ter 15% da representação dos bytes transmitidos em 1996. A partir de 1998, as requisições geradas dinamicamente tiveram um

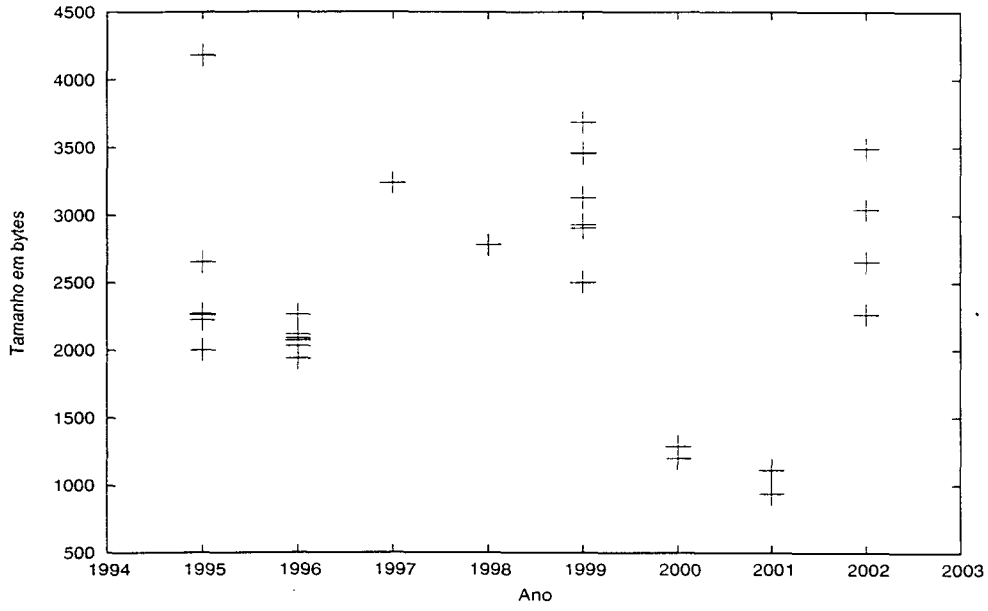


Figura 3.4: Evolução da mediana dos tamanhos dos arquivos em oito anos.

decréscimo acentuado chegando a ter representação próxima de 1%. Os demais tipos continuam com pequena representação e variação nos números de acessos. Os arquivos de áudio, vídeo e aplicações tiveram sua freqüência oscilando constantemente. Devemos observar que, mesmo com pequena representação dos acessos, estes arquivos têm fração de bytes elevada, ultrapassando, em alguns casos, os 10%.

3.5 Considerações Finais

Neste capítulo foram apresentadas as cargas de trabalho estudadas, sendo observadas algumas características para este conjunto de cargas, tais como:

- Mais de 98% das solicitações foram para o método GET, o que mostra que a Web é amplamente utilizada para consulta de informações;
- Os códigos de *status* 200 e 304, que resultam no sucesso para obtenção do arquivo solicitado pela requisição, somaram mais de 82% das requisições. O código 304 teve um crescimento

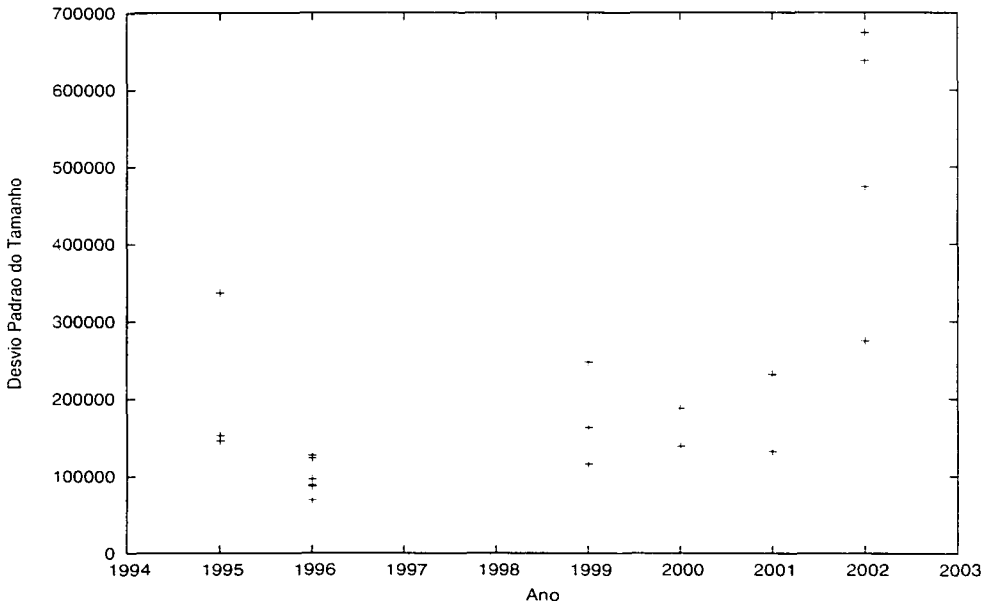


Figura 3.5: Evolução do desvio padrão dos tamanhos dos arquivos em oito anos.

acentuado em relação aos dados dos trabalhos anteriormente publicados. Isto indica que um número maior de arquivos é encontrado no cache;

- O tamanho médio dos arquivos cresceu pouco, a mediana diminuiu e o tamanho do maior arquivo cresceu muito, evidenciando um crescimento na variabilidade dos tamanhos transmitidos. Os arquivos do tipo html e imagens representam a maior parcela dos acessos, mas em uma porcentagem inferior à apresentada nos trabalhos referenciados. O tamanho médio das imagens diminuiu;

- A variabilidade dos tamanhos dos arquivos têm tendência de crescimento com o passar dos anos.

O próximo capítulo apresenta a correlação entre tamanho da resposta e o tempo de serviço. É detalhada a correlação entre as grandezas tamanho da requisição e tempo de serviço.

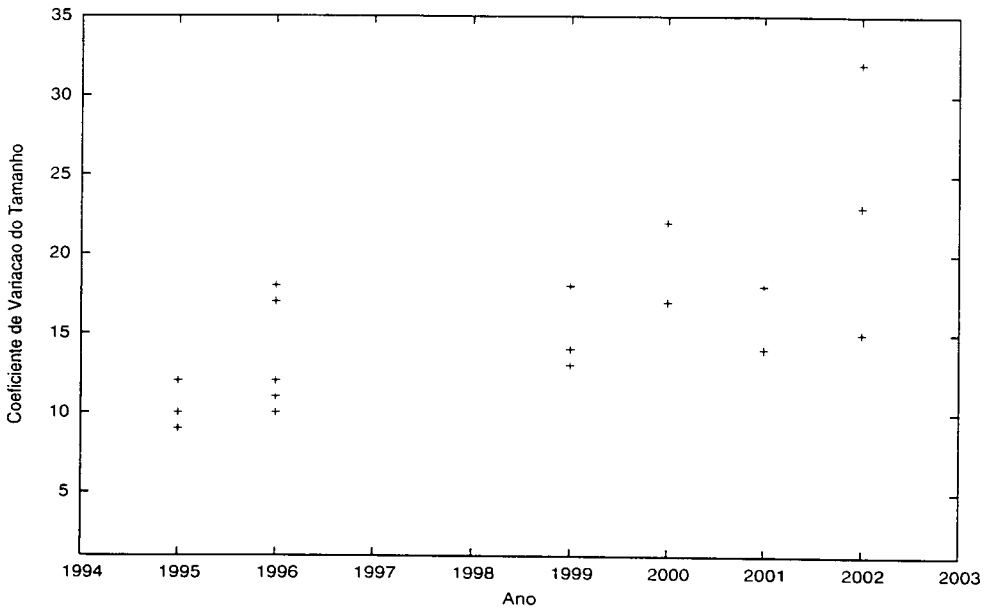


Figura 3.6: Evolução do coeficiente de variação dos tamanhos dos arquivos em oito anos.

Ano	Carga	Período	Registros	Menor	Maior	Média	Mediana	Desvio Padrão	COV
1995	BU(G) [1]	90	52.901	N/A	16.710	27.480	2.259	336.495	12,24
	BU(U) [1]	26	414.350	N/A	14.160	16.240	2.652	145.939	8,98
	KOREA [1]	24	1.681.963	N/A	56.220	14.640	2.222	152.902	10,44
	BL [33]	37	53.399	8	7.763	12.600	2.654	N/A	N/A
	BR [33]	38	179.600	18	9.797	56.074	1.999	N/A	N/A
	U [33]	190	173.597	1	18.384	11.927	2.268	N/A	N/A
	NASA [33]	28	1.385.259	28	3.341	19.323	4.179	N/A	N/A
1996	DEC1 [1]	1	1.304.565	N/A	22.620	8.792	2.087	86.758	9,86
	DEC2 [1]	1	1.293.147	N/A	31.690	8.598	1.939	88.312	10,27
	VT-LIB [1]	62	127.853	N/A	12.780	7.008	2.070	68.958	9,83
	VT-CS [1]	322	570.385	N/A	20.440	9.762	2.115	123.637	17,64
	VT-HAN [1]	131	440.345	N/A	26.860	8.272	2.261	95.956	11,60
	AUB [1]	2	19.259	N/A	6.779	8.863	1.938	96.405	10,87
	AOL [1]	1	883.082	N/A	46.290	7.545	2.030	127.056	16,83
1997	POP98 [33]	12	2.111.766	17	20.491	9.065	3.235	N/A	N/A
1998	PORTUGAL [33]	7	1.193.404	51	17.499	9.033	2.779	N/A	N/A
1999	POP99HUG [33]	4	1.121.747	17	26.527	10.527	2.929	N/A	N/A
	POP99ZEZ [33]	4	1.120.830	16	64.470	15.223	2.905	N/A	N/A
	NLANR-UC [33]	1	800.534	51	52.444	14.105	3.461	N/A	N/A
	NLANR-BO1 [33]	1	606.179	51	48.806	14.894	3.683	N/A	N/A
	USASK [30]	45	21.070.330	N/A	N/A	8.422	2.500	116.139	13,79
	CANARIE [30]	45	7.310.038	N/A	N/A	12.297	3.455	163.304	13,28
	NLANR-UC [30]	30	24.560.611	N/A	N/A	14.066	3.128	247.561	17,60
2000	NLANR-BO [36]	1	415.132	N/A	43.000	8.174	1.287	138.958	17
	NLANR-PB [36]	1	1.628.713	N/A	76.000	8.537	1.198	187.814	22
2001	NLANR-BO [36]	1	306.969	N/A	73.000	12.864	936	231.552	18
	NLANR-PB [36]	1	886.608	N/A	33.000	9.471	1.112	132.594	14
2002	NLANR-PB	4	3.322.450	106	281.000	19.845	2.263	474.603	23,91
	NLANR-SD	3	1.083.415	132	311.000	28.416	2.652	674.253	23,73
	NLANR-SV	4	1.851.472	162	156.000	17.694	3.491	275.059	15,54
	NLANR-RTP	3	4.015.680	158	377.000	19.896	3.041	637.952	32,07

Tabela 3.8: Evolução dos tamanhos das transferências registradas nos caches Web em oito anos.

Ano	Carga	HTML	Imagens	CGI	Áudio	Vídeo	Aplicações
1995	BU(G) [1]	12,8	82,8	2,1	0,1	0	0,2
	BU(U) [1]	12,5	84,8	1,5	0,1	0	0
	KOREA [1]	21,7	66,8	5,7	0,2	0,2	0,4
1996	DEC1 [1]	11,3	59,1	1,8	0,0	0,0	0,0
	DEC2 [1]	10,9	58,5	1,6	0,0	0,0	0,0
	VT-LIB [1]	15,0	67,1	12,7	0,0	0,0	0,0
	VT-CS [1]	20,6	51,3	19,2	0,2	0	0,1
	VT-HAN [1]	14,8	69,5	8,7	0,1	0,0	0,0
	AUB [1]	13,6	67,5	9,3	0,0	0,0	0,0
	AOL [1]	14,2	73,9	8,2	0,0	0,0	0,0
1998	WORLDCUP [4]	9,8	88,1	0,0	0,0	0,0	0,6
1999	USASK [30]	19,4	77,5	1,9	0,2	0,0	0,0
	CANARIE [30]	20,5	76,3	1,6	0,4	0,1	0,1
	NLANR-UC [30]	21,5	68,3	1,6	0,2	0,0	0,1
2002	NLANR	28,5	59,3	0	0,2	0,3	10,7

Tabela 3.9: Evolução dos acessos por tipo de arquivo em oito anos.

Ano	Carga	HTML	Imagens	CGI	Áudio	Vídeo	Aplicações
1995	BU(G) [1]	2,3	21,4	10,0	28,7	3,5	6,6
	BU(U) [1]	2,0	48,5	13,9	6,5	18,5	3,8
	KOREA [1]	2,7	48,8	7,7	2,1	12,5	14,4
1996	DEC1 [1]	1,0	38	8,0	0,0	0,0	0,0
	DEC2 [1]	1,0	40,1	7,5	0,0	0,0	0,0
	VT-LIB [1]	1,5	64,1	13,3	1,7	11,6	1,5
	VT-CS [1]	0,8	46	15,0	5,2	8,0	6,8
	VT-HAN [1]	1,1	53,9	9,3	7,5	13,7	2,8
	AUB [1]	1,5	52,1	7,8	0,1	27,4	1,0
	AOL [1]	7,9	59,9	8,2	1,5	5,0	2,2
1998	WORLDCUP [4]	38,6	35,0	0,3	0,0	0,0	20,3
1999	USASK [30]	23,2	52,1	0,8	3,3	10,2	3,2
	CANARIE [30]	17,8	49,0	0,2	5,7	12,3	5,1
	NLANR-UC [30]	18,1	39,9	1,0	3,3	6,3	7,8
2002	NLANR	23,9	21,8	0,0	3,8	12,1	36,5

Tabela 3.10: Porcentagem dos bytes transferidos por tipo de arquivo em oito anos.

Capítulo 4

Correlação entre Tamanho da Resposta e Tempo de Serviço

Neste capítulo é avaliada a correlação entre o tamanho da resposta a uma requisição HTTP e o tempo necessário para servi-la. Como discutido no capítulo 2, poucos trabalhos descrevem o comportamento dos tempos de serviço [34, 42]. Neste capítulo são apresentados dados estatísticos para os tamanhos das respostas e para os tempos de serviço, em classes de tamanhos e em conjuntos de registros. Na seção 4.1 são apresentados os dados estatísticos para os tamanhos das respostas e para os tempos de serviço. Na seção 4.2 avaliamos a correlação em faixas de tamanhos e por conjuntos de tipos das respostas. Na seção 4.3 são apresentadas as considerações finais do capítulo.

4.1 Dados Estatísticos para Tamanhos e Tempos

Neste trabalho a correlação entre o tamanho da resposta, dado em bytes, e o tempo necessário para enviá-la, dado em milissegundos, é avaliada em três conjuntos de registros: todos os registros, o conjunto de *hits* e o conjunto de *misses*. Assim podemos avaliar a influência dos *hits* e *misses* nos tempos de serviço das requisições.

As Figuras 4.1, 4.2, 4.3 e 4.4 apresentam gráficos do tamanho dos arquivos transferidos (eixo x) e do tempo de transmissão para cada transferência (eixo y). Os gráficos estão em escala

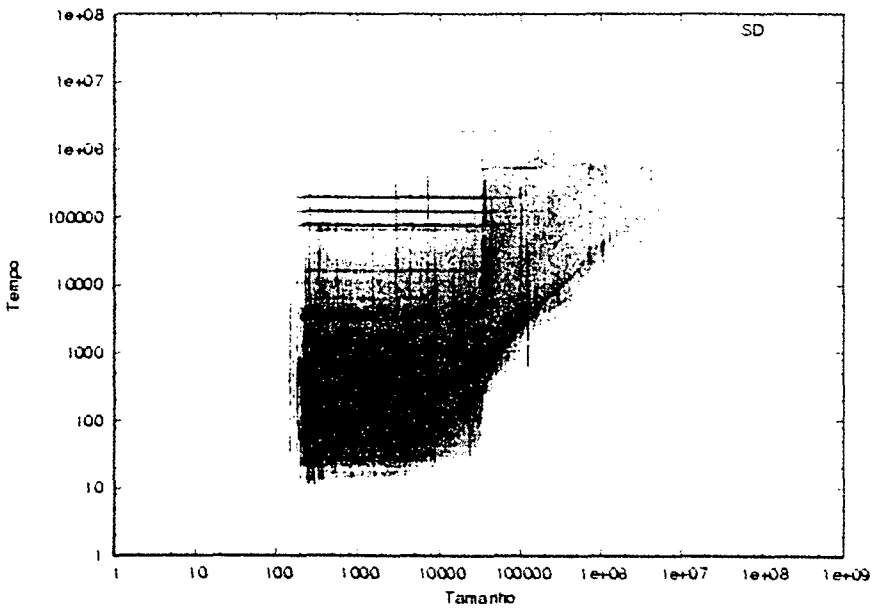
logarítmica nos dois eixos. Cada figura apresenta dados de um dos quatro caches. As requisições dos quatro dias estão agrupadas no mesmo gráfico. Os gráficos rotulados com (a) nas figuras referem-se às requisições *misses* enquanto os gráficos rotulados com (b) das mesmas figuras são as requisições *hits*.

Todos os gráficos mostram grande variabilidade nos tempos de serviço para respostas de mesmo tamanho, em especial quando as respostas são de tamanho pequeno. Esta variação chega a cinco ordens de grandeza. As "nuvens" apresentam uma cauda que sugere que os tempos estão correlacionados com os tamanhos, para tamanhos maiores. Observamos que a grande maioria das requisições é para arquivos com menos de 50 KBytes (mais de 90% das requisições). Com o crescimento do tamanho das requisições é possível perceber que a densidade de pontos diminui, pois o número de requisições para arquivos maiores é menor. Uma explicação para esta variabilidade nos tempos de serviço é a combinação dinâmica da variabilidade dos tamanhos das respostas e das diferentes capacidades de conexões entre os clientes e os caches, além dos atrasos gerados pelo tráfego [34].

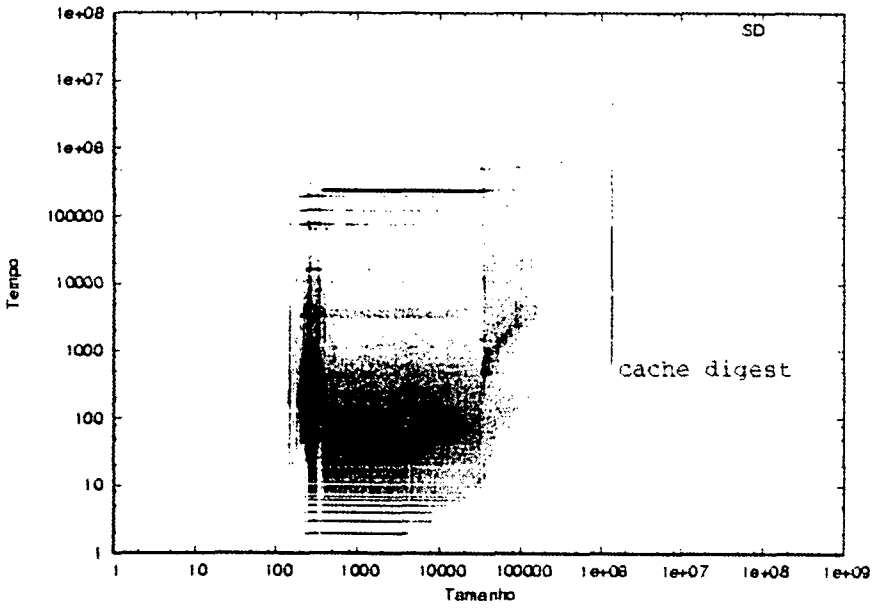
Comparando os gráficos das requisições *hits* com os das requisições *misses*, podemos observar claramente que os tempos das respostas mais rápidas para os *hits* são cerca de uma ordem de grandeza menores do que os tempos das respostas mais rápidas para os *misses*. Isto ocorre porque, no caso de *miss*, a necessidade do estabelecimento de conexão com o servidor (desnecessárias no caso de *hit*) prolonga o tempo de serviço.

Nos gráficos das requisições *hits* há uma ocorrência maior de um tipo de registro com tamanho próximo a 1 Mbyte, estas ocorrências são identificadas nos gráficos como cache *digest*. Trata-se de um sumário do conteúdo do cache de um servidor que é trocado periodicamente com outros servidores pertencentes à hierarquia. Transfere-se uma lista das entradas de um cache para o outro, para que as pesquisas sejam mais eficientes [41].

A Tabela 4.1 apresenta os valores da média (μ), do desvio padrão (σ) e do coeficiente de variação (σ/μ) para os tamanhos das respostas e os tempos de serviço medidos para estas respostas. Na última coluna é apresentado o número total de registros, bem como o número de *hits* e de *misses*. Avaliando os dados apresentados na última coluna é possível calcular o HR (*hit ratio*), que é a fração das requisições atendidas pelo cache, que variou entre 19% e 33%.

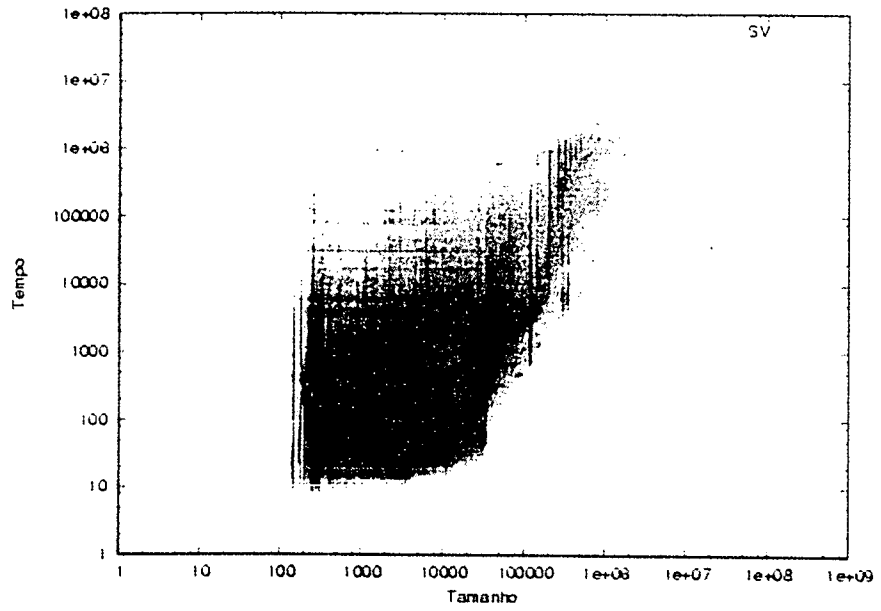


(a) cache SD *misses*

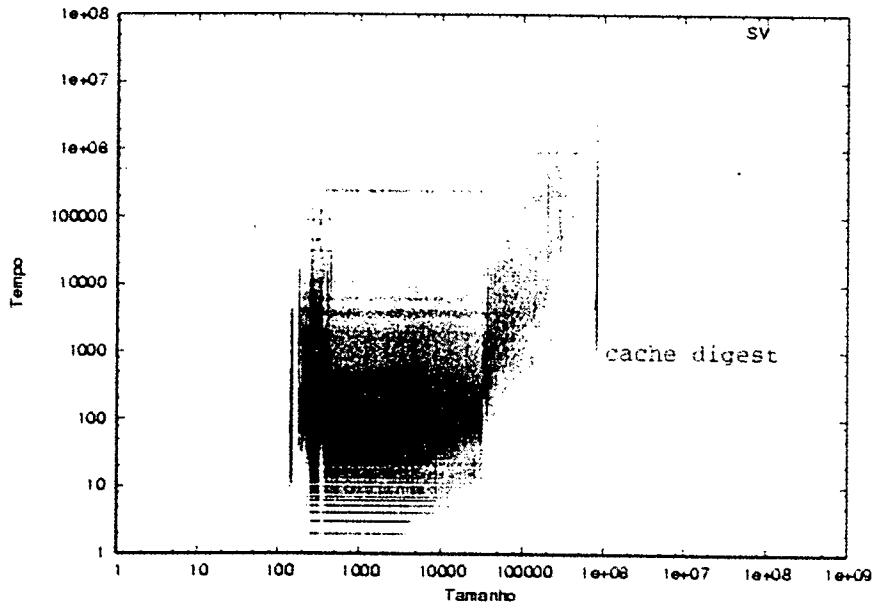


(b) cache SD *hits*

Figura 4.1: Tempo de transmissão (em milissegundos) em função do tamanho para requisições (em bytes) do cache SD *hits* (a) e *misses* (b).

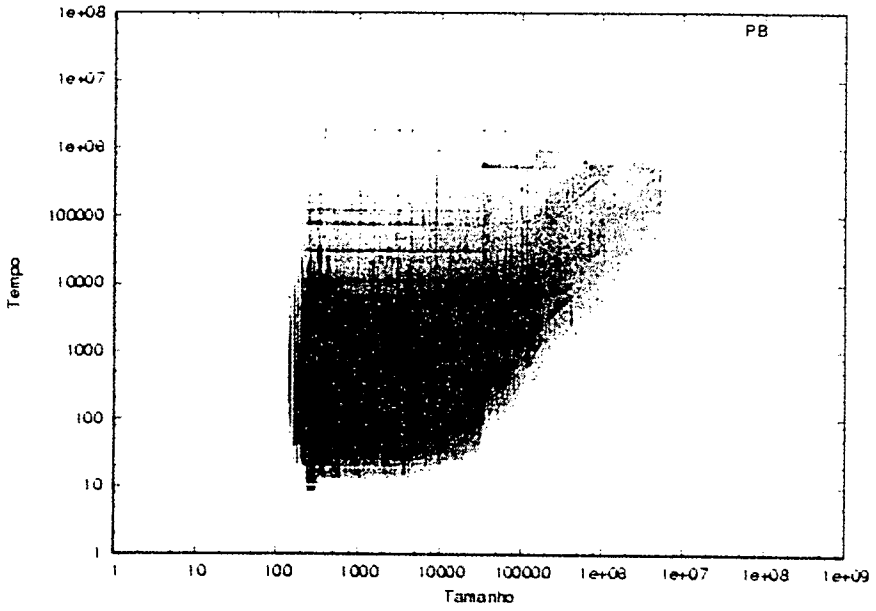


(a) cache SV *misses*

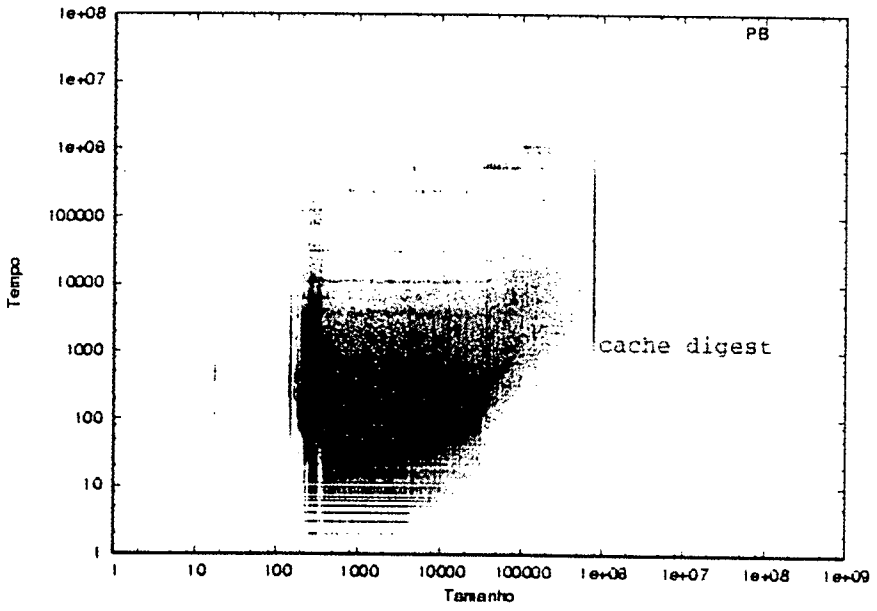


(b) cache SV *hits*

Figura 4.2: Tempo de transmissão (em milissegundos) em função do tamanho para requisições (em bytes) do cache SV *hits* (a) e *misses* (b).

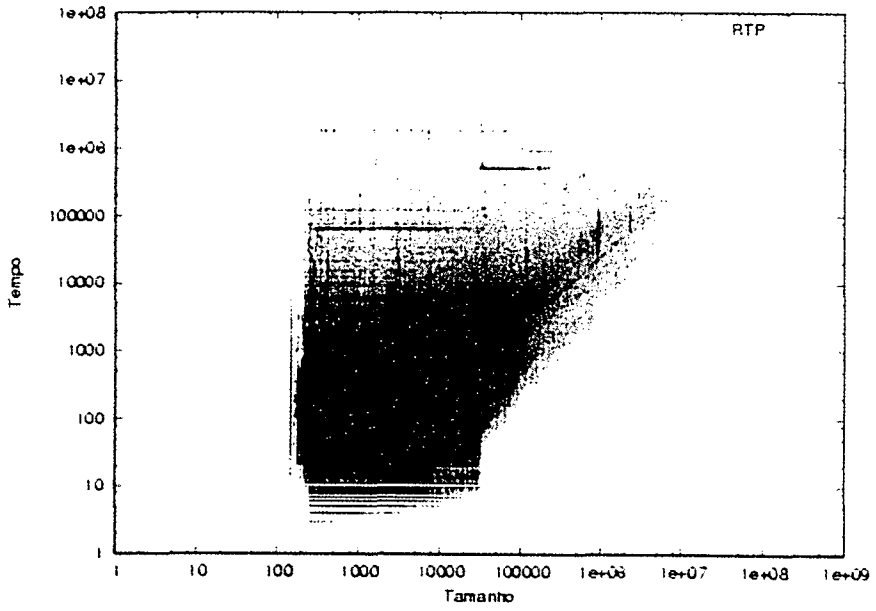


(a) cache PB *misses*

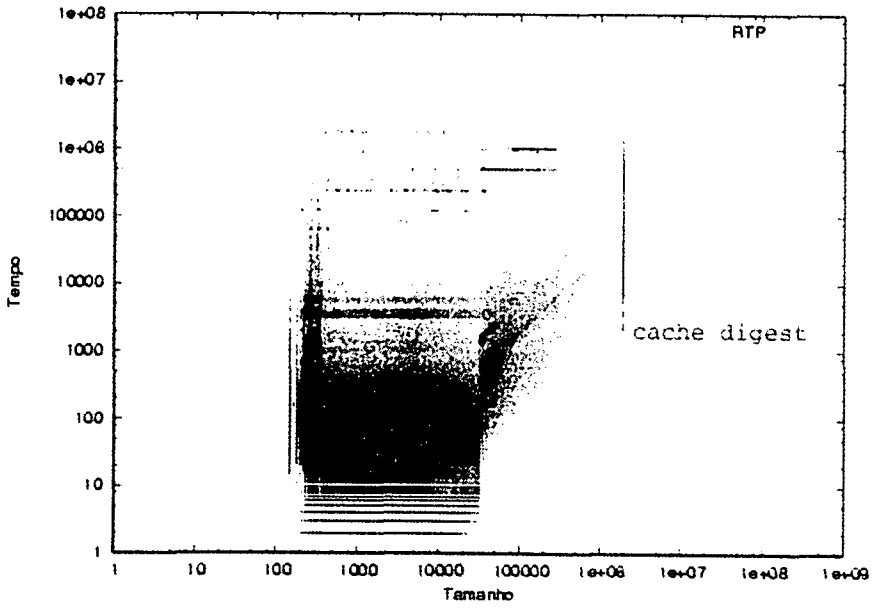


(b) cache PB *hits*

Figura 4.3: Tempo de transmissão (em milissegundos) em função do tamanho para requisições (em bytes) do cache PB *hits* (a) e *misses* (b).



(a) cache RTP *misses*



(b) cache RTP *hits*

Figura 4.4: Tempo de transmissão (em milissegundos) em função do tamanho para requisições (em bytes) do cache RTP *hits* (a) e *misses* (b).

Avaliando os dados obtidos para os tamanhos é possível perceber que a média e o desvio padrão dos tamanhos dos arquivos para as requisições *hits* são sempre menores do que para as requisições *misses*. As médias das requisições *hits* ficaram entre 5,58 e 15,33 Kbytes sendo que, para as requisições *misses*, a média esteve entre 12,90 e 21,08 Kbytes. As médias para o conjunto "*todos*" ficou entre 12,15 e 19,67 Kbytes. Neste conjunto, as médias dos tamanhos ficaram mais próximas das médias obtidas para os *misses* pois estes têm influência dominante. As requisições *hits* têm tamanhos menores devido às requisições do código 304, que, como apresentado no capítulo 3, têm respostas pequenas, e aos algoritmos de substituição de objetos, que mantém preferencialmente arquivos pequenos nos caches. Os dados disponíveis na literatura referentes aos tamanhos médios reportam tamanhos entre 7,01 e 12,73 Kbytes para os *hits* e tamanhos entre 9,76 e 16,63 Kbytes para os *misses* [34].

Os desvios padrão dos tamanhos para as requisições *hits* ficaram entre 77,12 e 348,43 Kbytes sendo que, para as requisições *misses*, o desvio padrão ficou entre 265,72 e 611,12 Kbytes. Os desvios padrão para o conjunto "*todos*" ficaram entre 224,67 e 553,47 Kbytes. Os dados obtidos com o desvio padrão mostram a grande variação dos tamanhos dos arquivos transmitidos através dos caches Web. Avaliando os dados caracterizados anteriormente é possível verificar que os desvios padrão dos *hits* aumentaram, sendo que anteriormente variavam entre 40,58 e 118,0 Kbytes. O mesmo ocorreu para as requisições *misses*, pois anteriormente variavam entre 75,69 e 648,2 Kbytes [34].

As estatísticas para os tempos mostram que a média e o desvio padrão dos tempos de serviço para as requisições *hits* são menores que os apresentados pelos *misses*, a exceção é o cache PB que apresenta tempo médio e desvio padrão maiores para as requisições *hits*, embora a diferença seja pequena. As médias dos tempos ficaram entre 1,53 e 6,90 segundos para os *hits* e, para as requisições *misses*, ficaram entre 1,88 e 8,34 segundos. Para o conjunto "*todos*" a média ficou entre 1,78 e 7,99 segundos. Comparando com os dados disponíveis na literatura é possível verificar que os tempos médios das requisições *hits* aumentaram, porque antes variavam entre 0,55 e 3,49, e que os tempos médios para as requisições *misses* diminuiram, pois estes variavam entre 5,09 e 13,53 segundos. Uma possível explicação para este fato é a melhoria da capacidade dos *backbones*, que contribui para a diminuição dos tempos dos *misses*. Quanto aos tempos dos

hits. seu aumento pode ser explicado pela obsolescência do hardware do cache, conjugado com um aumento no número de requisições atendidas.

Os desvios padrão dos tempos para as requisições *hits* ficaram entre 32.02 e 63.57 segundos. sendo que, para as requisições *misses*. o desvio padrão ficou entre 49,41 e 85,77 segundos. Os desvios padrão para o conjunto "*todos*" ficou entre 48,08 e 80,90. Os dados obtidos com o desvio padrão mostram grande variação dos tempos de serviço. Um conjunto de fatores pode ser considerado como prováveis responsáveis por esta variação nos tempos de serviço, entre eles a variabilidade nas conexões dos clientes, a variação dos tamanhos das respostas e o custo inicial do estabelecimento de uma conexão TCP. O custo da conexão é amortizado somente em respostas maiores, pois estas requerem mais pacotes para sua transmissão.

Cache	tipo	Tamanho (KB)			Tempo (s)			N° req.
		μ	σ	σ/μ	μ	σ	σ/μ	
PB	*hit*	10,93	94,63	8,65	2,57	53,63	20,86	994.710
	miss	12,90	410,55	31,82	2,25	49,41	21,96	4.184.044
	todos	12,52	371,34	29,65	2,31	50,25	21,75	5.178.754
RTP	*hit*	5,58	77,12	13,82	1,56	34,88	22,35	1.893.630
	miss	16,40	611,12	37,26	1,88	53,03	28,20	4.144.376
	todos	13,01	508,17	39,05	1,78	48,08	26,98	6.038.006
SD	*hit*	15,33	348,43	22,72	6,90	63,57	9,21	375.543
	miss	21,08	605,20	28,70	8,34	85,77	10,28	1.158.615
	todos	19,67	553,47	28,13	7,99	80,90	10,12	1.534.158
SV	*hit*	9,45	99,80	10,56	1,53	32,02	20,92	879.651
	miss	13,50	265,72	19,68	5,34	79,78	14,94	1.769.921
	todos	12,15	224,67	18,49	4,07	67,79	16,65	2.649.572

Tabela 4.1: Dados estatísticos obtidos nos processamentos.

Pode-se observar que o comportamento é consistente para todas as cargas analisadas. Os tamanhos são mais variáveis do que os tempos de serviço para o conjunto "*todos*", o que pode ser observado pelo coeficiente de variação. Os coeficientes de variação do conjunto de *hits* ficaram

entre 8,65 e 22,72, para os tamanhos, e entre 9,21 e 22,35 para os tempos. Comparando com dados obtidos na literatura é possível verificar que o coeficiente de variação para o tempo diminuiu, pois estava entre 19,54 e 58,53 [34], e que coeficiente de variação para os tamanhos aumentou, pois estava entre 5,78 e 11,89. Uma possível justificativa para estas mudanças é que os tamanhos têm coeficiente de variação maiores devido ao crescimento dos arquivos grandes e principalmente pela popularização destes arquivos. Os arquivos de áudio, vídeo e de programas compactados já podem ser obtidos em um tempo razoável pelos usuários da Web, pois as conexões estão cada vez mais velozes. A possível justificativa para a diminuição do coeficiente de variação dos tempos dos *hits* é a melhoria da capacidade das redes disponíveis para os usuários. Os coeficientes de variação para o conjunto de *misses* variaram entre 19,68 e 37,26, para os tamanhos, e entre 10,28 e 28,20 para os tempos. Comparando com os dados publicados anteriormente, percebe-se que os coeficientes de variação dos *misses* aumentaram. Anteriormente os coeficientes de variação para os *misses* variavam entre 7,75 e 42,35 para os tamanhos e entre 5,46 e 13,53 para os tempos. Uma possível justificativa para essa variação nos tamanhos é o fato de que os arquivos maiores não são constantemente armazenados nos caches e geralmente necessitam serem solicitados dos servidores, gerando um maior coeficiente de variação. Para os tempos, a justificativa é o fato de que o caminho a ser seguido pela requisição, atravessa um número maior de *hosts*, como já descrito anteriormente, quanto maior o número de *hosts* maior a probabilidade de atrasos e de aumento do coeficiente de variação.

Como já descrito na seção 3.3, as medianas dos tamanhos variaram entre 2,21 e 3,41 Kbytes, ficando bem abaixo da média, que variou entre 17,28 e 27,75 Kbytes, resultados semelhantes aos apresentados em [2]. O coeficiente de variação (σ/μ) dos tamanhos variou de 8,65 a 39,05, mostrando a grande variabilidade do conjunto de requisições. Estas características, a saber, mediana menor do que a média e grande coeficiente de variação, indicam que uma distribuição de cauda pesada representa os dados. Como a maioria das requisições é de *misses*, o conjunto de *misses* tem influência dominante no conjunto de todas as requisições. Os modelos para descrever as variáveis tempos de serviço e tamanhos das respostas serão discutidos no Capítulo 5.

4.2 Avaliação da Correlação por Faixas de Tamanhos

Para consolidar as observações a respeito da correlação apresentada nas Figuras 4.1, 4.2, 4.3 e 4.4 foram realizados vários processamentos nos registros dos quatro caches. Inicialmente os registros foram divididos em quatro classes para enfocar as características de arquivos que sejam similares em tamanho. Foram considerados na classe 1 os arquivos menores ou iguais a 2 Kbytes na classe 2 os arquivos maiores do que 2 Kbytes e menores ou iguais a 30 Kbytes, na classe 3 os arquivos maiores do que 30 Kbytes e menores ou iguais a 500 Kbytes, e na classe 4 os maiores que 500 Kbytes. Na classe 1 foram agrupados 62,8% dos registros, na classe 2 ficaram 32,3% dos registros, a classe 3 foi responsável por 4,5% dos registros e a classe 4 por 0,3% dos registros. Posteriormente foram processados os arquivos por tipo de resposta. Os arquivos foram divididos em *hits*, *misses* e todos os arquivos, sendo distribuídos nas quatro classes descritas.

A divisão de registros em faixas de tamanho já foi apresentada em outros trabalhos [33, 42]. Em um deles [42], os registros foram divididos em três faixas de tamanho, sendo considerados como MICE os arquivos menores que 30 Kbytes, ELEPHANTS os arquivos maiores ou iguais a 30 Kbytes e menores ou iguais a 500 Kbytes, e MAMMOTHS os arquivos com mais de 500 Kbytes. O problema da distribuição usada é que quase todos os arquivos (94%) ficam na primeira classe dos tamanhos, e uma avaliação entre classes de arquivos similares fica prejudicada.

Para a obtenção dos valores da correlação (r) entre o tamanho da resposta (x) e o tempo de serviço (y), para n observações, foi utilizada a seguinte fórmula:

$$r = \frac{\Sigma\frac{xy}{n} - \frac{\Sigma x \Sigma y}{n}}{\frac{\sqrt{n\Sigma(x)^2 - (\Sigma(x))^2}}{n(n-1)} * \frac{\sqrt{n\Sigma(y)^2 - (\Sigma(y))^2}}{n(n-1)}}$$

onde x e y são as variáveis cuja correlação será avaliada.

A Tabela 4.2 apresenta as correlações obtidas entre o tamanho da resposta e o tempo de serviço. Na primeira coluna estão os caches, na segunda os tipos de respostas, nas quatro colunas subseqüentes as classes em que foram avaliadas a correlação, e na última coluna a correlação para todos os registros. A correlação para as classes 1 e 2 variou entre 0 e 0,10. Esta fraca correlação para arquivos pequenos pode ser justificada por vários fatores, tais como as condições da rede e a capacidade das conexões. A transmissão de arquivos pequenos é fortemente influenciada pela

latência da rede, porque geralmente consiste na transmissão de poucos pacotes e a maior parte do tempo é gasto no estabelecimento de conexões de rede do que na trasmissão de pacotes. A ocorrência de um evento inesperado em transferências pequenas, tal como a perda de um pacote, pode aumentar o tempo de transmissão em pelo menos uma ordem de grandeza [23].

Cache	tipo	Correlação por Classe				
		Classe 1	Classe 2	Classe 3	Classe 4	Todos
SD	*hit*	0,10	0,02	0,18	0,29	0,28
	miss	0.01	0,02	0.22	0,37	0,37
	todos	0,04	0,01	0,22	0,37	0,36
RTP	*hit*	0,02	0,01	0,15	0,22	0,21
	miss	0,00	0,02	0,11	0,57	0,52
	todos	0,01	0,01	0,11	0,56	0,48
SV	*hit*	0,01	0,02	0,47	0,36	0,39
	miss	0,03	0,04	0,52	0,38	0,42
	todos	0,03	0,04	0,51	0,39	0,42
PB	*hit*	0,00	0,01	0,04	0,21	0,14
	miss	0,01	0,02	0,11	0,55	0,40
	todos	0,01	0,02	0,09	0,55	0,36

Tabela 4.2: Correlação tamanho X tempo de resposta dos processamentos.

Os resultados para as classes 3 e 4 mostram que a correlação entre o tamanho e o tempo de serviço aumenta significativamente, para tamanhos maiores, sendo que na classe 3 a correlação variou entre 0,04 e 0,52, e na classe 4 esteve entre 0,21 e 0,57. Em [42] a correlação para as classes de arquivos com tamanhos similares aos das classes 3 e 4 foram 0,33 e 0,53 respectivamente.

Outro fator a ser analisado é a correlação observada nos conjuntos de *hits* e *misses*. A correlação em caso de *misses* apresenta números significativamente maiores nas classes 3, 4 e todos. Isto pode ser explicado pelo fato de que os caches apresentam mais *hits* para arquivos pequenos, e mais *misses* para arquivos grandes. Além disso, o tamanho dos *hits* é menor do que o tamanhos dos *misses*, conforme apresentado na Tabela 4.1, e a correlação para arquivos

grandes é maior.

Avaliando os dados de todos os registros percebe-se o domínio das classes 3 e 4 sobre as demais classes e, portanto, os índices obtidos ficam próximos dos pertencentes às classes dominantes.

4.3 Considerações Finais

Neste capítulo as observações foram concentradas na correlação entre tamanho da resposta e o tempo de serviço. As principais observações são as seguintes:

- O HR variou entre 19 e 33%, esta taxa vêm diminuindo com o passar do tempo, e essa diminuição se deve ao crescimento progressivo da Web, ao número cada vez maior de arquivos, ao crescimento do número de usuários, ao aumento na diversidade dos acessos e ao surgimento ininterrupto de novas facilidades de uso e de novos serviços;

- A variabilidade nos tempos de serviço para respostas do mesmo tamanho, em especial para respostas de tamanho pequeno, chega a cinco ordens de grandeza. Mostramos que o coeficiente de variação para o tempo diminuiu e o coeficiente de variação para os tamanhos aumentou. Uma possível justificativa para estas mudanças é que os tamanhos têm coeficiente de variação maiores devido ao crescimento dos arquivos grandes e, principalmente, pela sua popularização. Os arquivos de áudio, vídeo e de programas compactados já podem ser obtidos em um tempo razoável para os usuários, pois as conexões estão cada vez mais velozes. A possível justificativa para a diminuição do coeficiente de variação dos tempos é o aumento da velocidade das conexões e a eficiência dos caches Web;

- Em arquivos com mais de 30 Kbytes existe forte correlação entre o tamanho da resposta e o tempo de serviço. Em arquivos pequenos a correlação é pequena ou inexistente. O principal motivo para a fraca correlação nos arquivos pequenos é o custo inicial do protocolo para cada conexão. Este custo é atenuado apenas em arquivos que precisem transmitir um número maior de pacotes para completar a requisição.

No próximo capítulo são analisados os modelos estatísticos para o tamanho da resposta e o tempo de serviço e são propostos modelos estatísticos para os tempos de serviço.

Capítulo 5

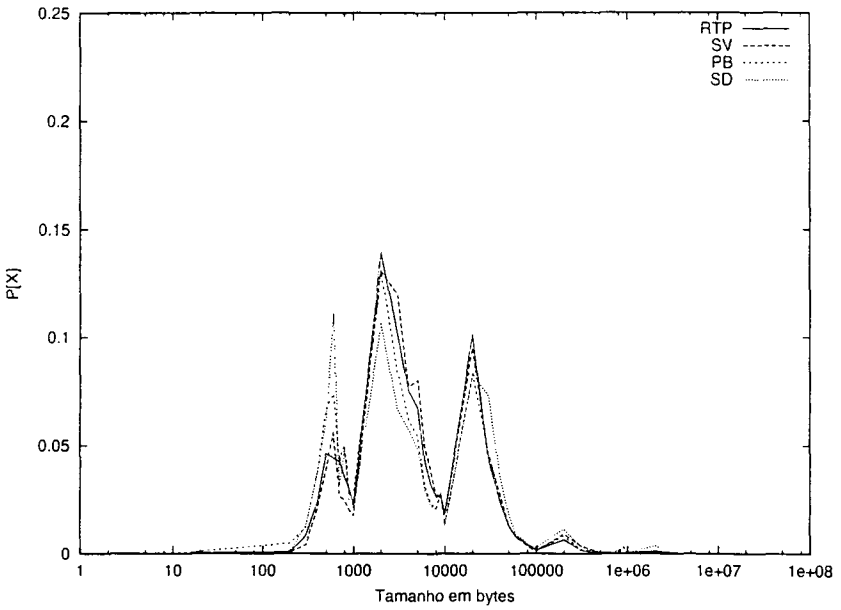
Modelo de Tempo de Serviço

Neste capítulo são analisados os modelos estatísticos para o tamanho e o tempo de transmissão e são propostos modelos estatísticos para os tempos de transmissão. Inicialmente apresentamos gráficos para as freqüências simples e acumulada dos tamanhos das respostas e dos tempos de serviço e posteriormente são apresentados os modelos estatísticos.

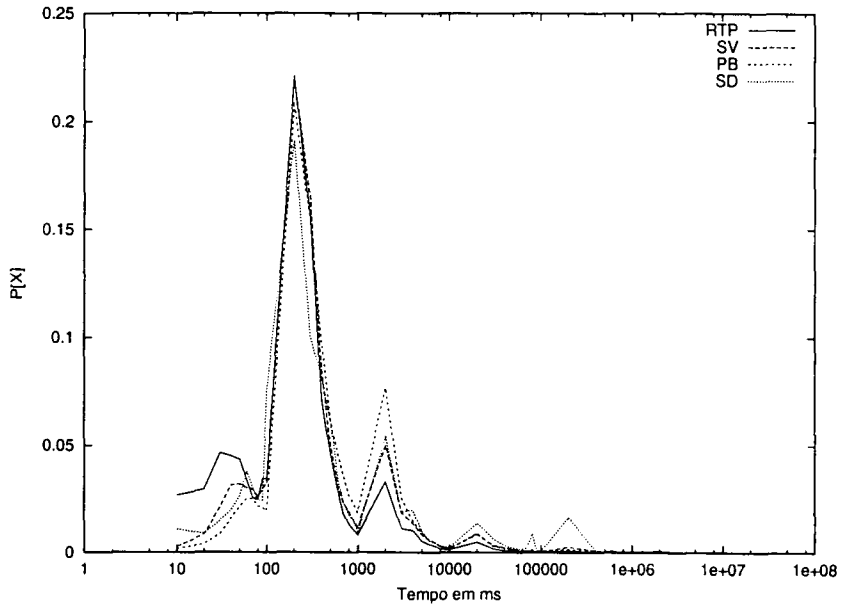
5.1 Distribuição de Freqüência das Requisições

O primeiro passo para a construção do modelo é avaliar as freqüências dos tamanhos e dos tempos comparativamente. Nesta seção as freqüências são plotadas em vários gráficos para os conjuntos de *hits*, *misses* e "*todos*". O objetivo é modelar o comportamento dos tamanhos das respostas e os tempos de serviço. A motivação é verificar se os tamanhos podem representar os tempos de serviço.

As Figuras 5.1, 5.2 e 5.3 apresentam a freqüência dos tamanhos (gráficos (a)) e dos tempos de serviço (gráficos (b)) para as duas variáveis em questão. O objetivo, em cada gráfico, não é identificar cada curva mas comparar seu comportamento. Verificamos que todas apresentam comportamento bastante similar. Os tamanhos estão distribuídos por uma faixa maior de valores, enquanto os tempos estão concentrados em uma faixa mais estreita. Além disso, os picos observados nos tempos são mais acentuados do que os observados nos tamanhos, caracterizando maior concentração de freqüência em faixas mais estreitas.

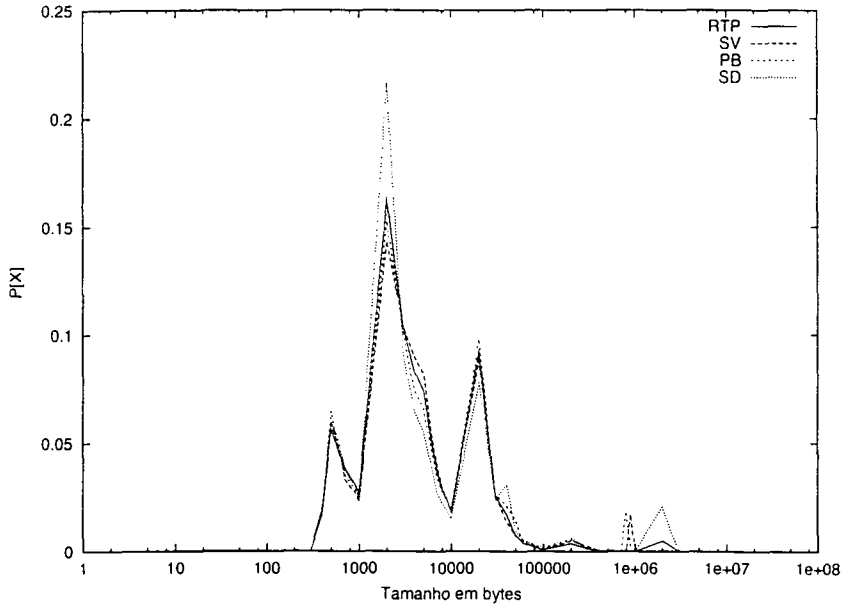


(a) Todas as requisições

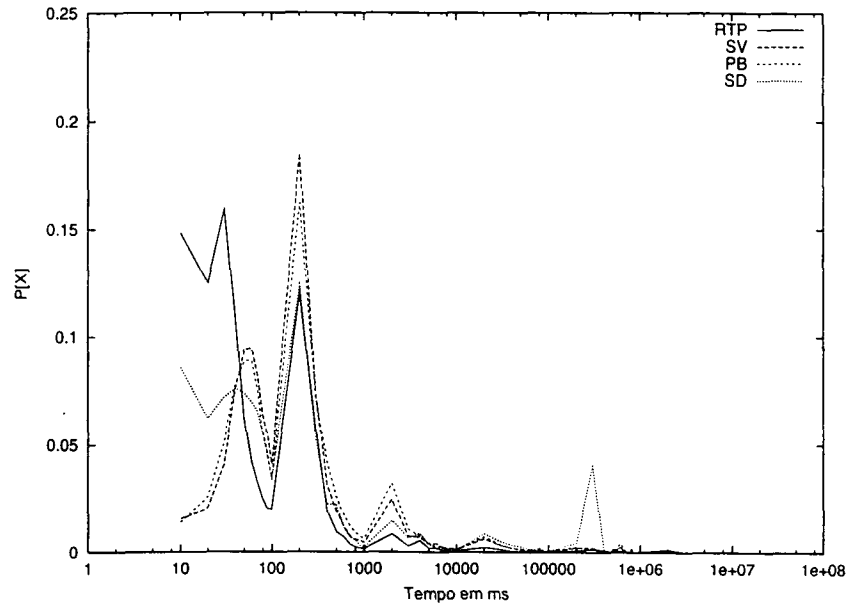


(b) Todas as requisições

Figura 5.1: Distribuição dos tamanhos das requisições (a) e tempos de serviço (b) para os diversos caches.

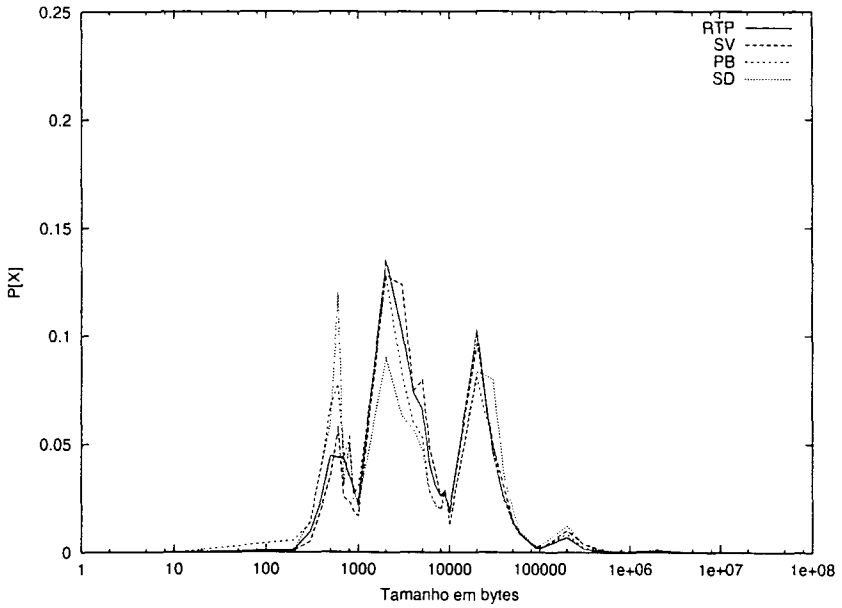


(a) Requisições *hits*

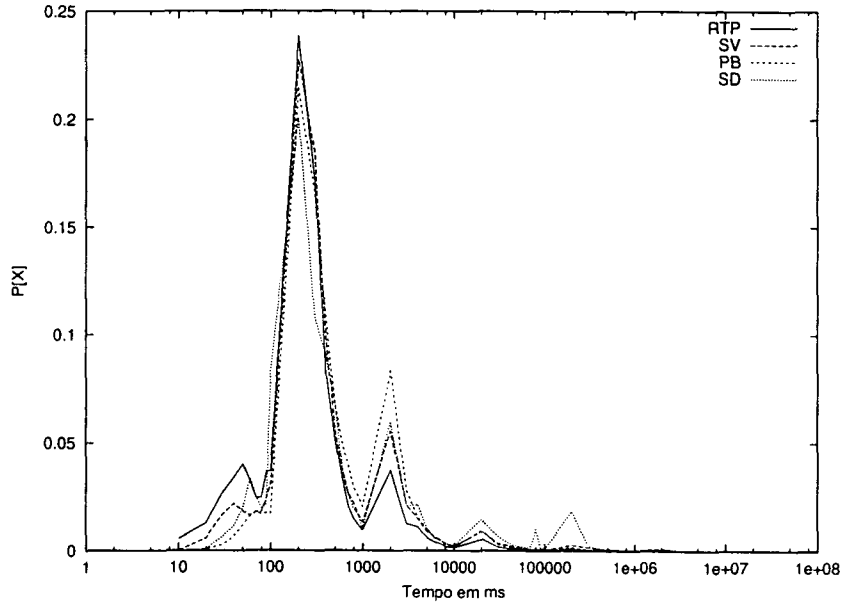


(b) Requisições *hits*

Figura 5.2: Distribuição dos tamanhos das requisições *hits* (a) e tempos de serviço *hits* (b) para os diversos caches.



(a) Requisições *misses*



(b) Requisições *misses*

Figura 5.3: Distribuição dos tamanhos das requisições *misses* (a) e tempos de serviço *misses* (b) para os diversos caches.

É possível verificar nos gráficos que, em alguns pontos, a freqüência de requisições é bem superior as demais. Nos gráficos dos tamanhos percebe-se picos nos pontos 500, 600, 2.000 e 20.000 bytes tanto para *hits* quanto para *misses*. As freqüências apresentadas nestes gráficos foram calculadas em intervalos (*bins*). Foram considerados dez intervalos para cada ordem de grandeza. Por exemplo, entre 10^3 e 10^4 temos os intervalos 1001:2000, 2001:3000, 3001:4000, até 9001:10000.

É possível verificar que, nas concentrações, as probabilidades dos tamanhos para os conjuntos *misses*, comparando com as probabilidades dos tamanhos dos *hits*, crescem em representação de acordo com o crescimento do tamanho. A concentração de referências nos gráficos dos tempos também é perceptível. Para os tempos foram usados os mesmos intervalos (*bins*) descritos no parágrado anterior. Percebe-se picos nos pontos 10, 30, 200, 300 e 2.000 milissegundos. No caso de *hits* a concentração nos tempos pequenos é bem acentuada.

A freqüência acumulada dos valores é apresentada no gráfico (a) das Figuras 5.4, 5.5 e 5.6. O complemento da freqüência acumulada representa a cauda da distribuição. As caudas para os três conjuntos de registros são apresentadas nos gráficos (b) das mesmas Figuras. Novamente o objetivo não é identificar cada curva mas verificar o comportamento dos conjuntos de dados, tempos e tamanhos.

Analisando os gráficos (a) das Figuras 5.4, 5.5 e 5.6 observamos que as curvas de tamanhos e tempos apresentam comportamento similar, porém estão deslocadas em relação ao eixo x em pelo menos uma ordem de grandeza. As curvas mais próximas do eixo y são as do tempo de serviço, para cada uma das quatro cargas, enquanto as curvas mais distantes do eixo y representam os tamanhos. A distância entre os conjuntos de curvas de tamanho e tempo é maior para o conjunto de hits e menor para o conjunto de misses.

Não é do nosso conhecimento nenhuma proposta de modelo para os tempos de serviço. Sem uma descrição do comportamento dos tempos de serviço, os trabalhos sobre políticas de escalonamento em caches e em servidores [16, 35] e os trabalhos sobre políticas de distribuição de tarefas em conjuntos de servidores [8, 22] assumem que a duração da transmissão é proporcional ao tamanho do arquivo transmitido. Estes gráficos mostram que os tempos são claramente distintos dos tamanhos. Portanto, a assunção de que o tempo de serviço pode ser modelado pelo

tamanho do arquivo transmitido pode ser invalidada.

As caudas das distribuições são aparentemente similares, e se confundem para valores acima de 10^5. As estimativas do parâmetro α para a distribuição de *Pareto*[1] utilizando o programa `aest` [17] revelaram valores próximos de 1 para todas as curvas. Esses valores indicam a presença de uma cauda pesada que prolonga-se à direita, portanto, existe uma probabilidade não desprezível de ocorrência de números muito grandes. Pode ser verificada existência de uma variabilidade extrema. A variabilidade cresce muito à medida que α decresce [33].

As Figuras 5.7 e 5.8 mostram os gráficos obtidos através do programa `aest` [17]. Foram criados quatro gráficos, um para cada cache. Nos gráficos são mostradas as concentrações de referências e os pontos em dez níveis definidos pelo programa. Estes níveis apresentam agregações sucessivas do conjunto de dados ao longo da porção da cauda julgada pelo *aest* como sendo a região de maior influência da cauda naquele nível. O gráfico também apresenta os valores calculados para o α de cada cache, bem como o número de pontos utilizados na estimativa.

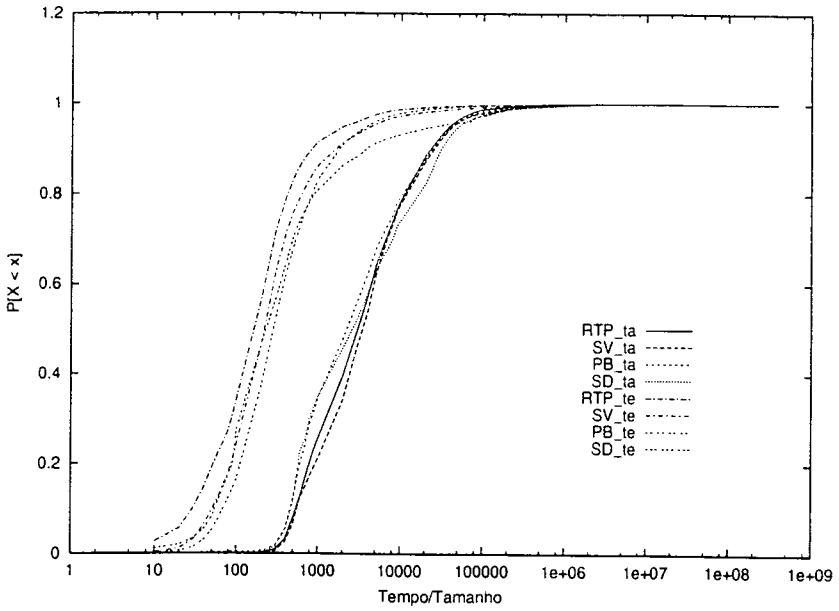
A carga de serviço imposta aos caches Web é bastante desigual, devido principalmente à distribuição dos tamanhos de arquivos. O cache Web pode tratar arquivos multimídia enormes e também arquivos com poucos bytes [5]. A caracterização dos tamanhos dos arquivos Web pode ser modelada por uma distribuição de cauda pesada. Além da variabilidade em relação ao tamanho dos arquivos servidos, o cache experimenta também efeitos relativos à variabilidade da transmissão dos arquivos para os clientes.

5.2 Modelo *Lognormal* para os Tempos de Serviço e Tamanhos das Respostas

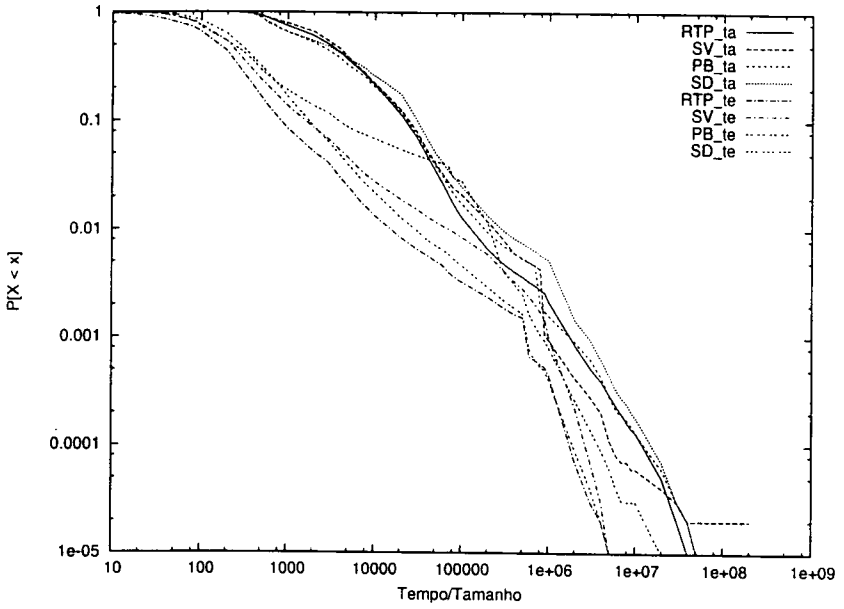
A partir dos gráficos apresentados na seção 5.1, observamos que as principais diferenças entre os conjuntos de tempos e tamanhos estão no corpo da distribuição e não na cauda. As curvas sugerem que a distribuição modelo poderia ser a mesma, porém seus parâmetros devem ser claramente distintos.

Como o objetivo é modelar o corpo da distribuição, descartamos a distribuição de *Pareto*,

[1] Descrito na seção 2.6.2, página 30

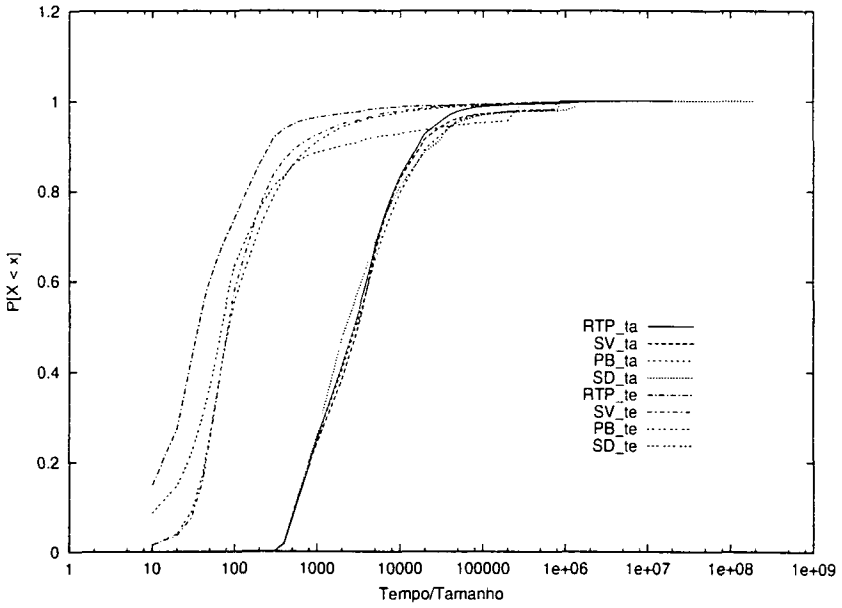


(a) Distribuição acumulada de todas as requisições

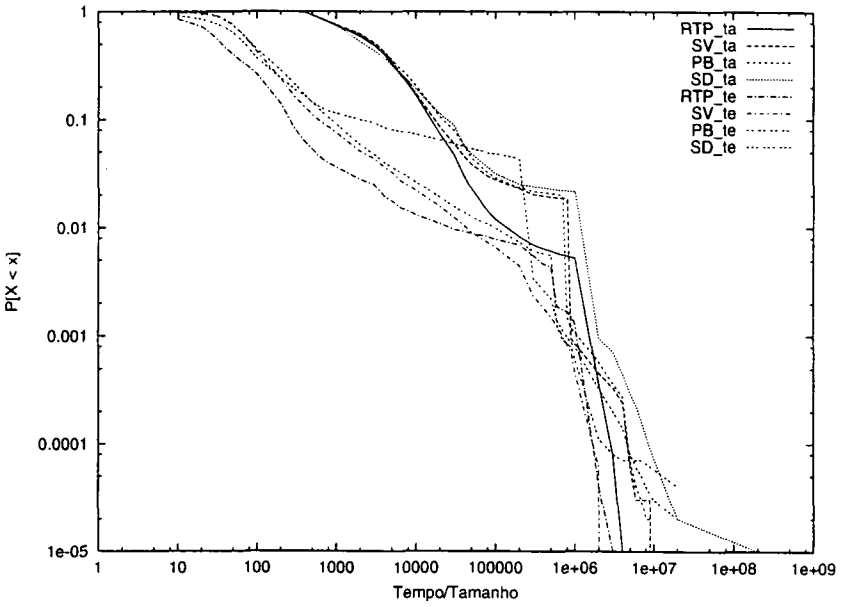


(b) Cauda das distribuições de todas as requisições

Figura 5.4: Distribuição acumulada (a) e cauda das distribuições (b) dos tamanhos das requisições e dos tempos de serviço.

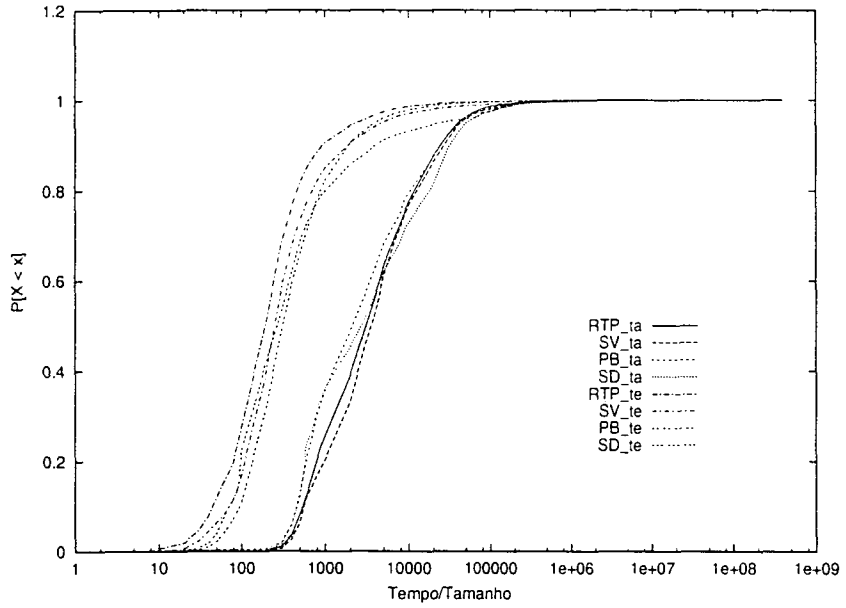


(a) Distribuição acumulada das requisições *hits*

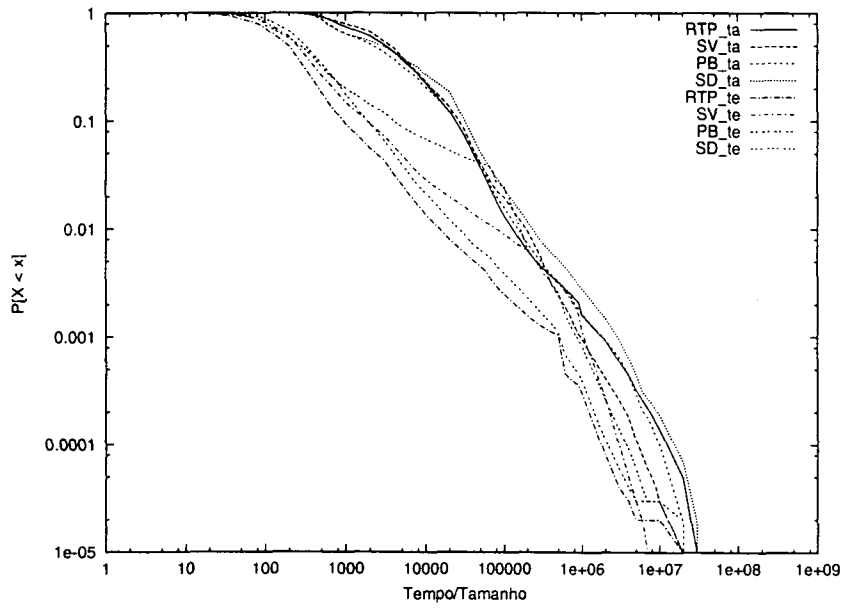


(b) Cauda das distribuições das requisições *hits*

Figura 5.5: Distribuição acumulada (a) e cauda das distribuições (b) dos tamanhos das requisições e dos tempos de serviço dos *hits*.

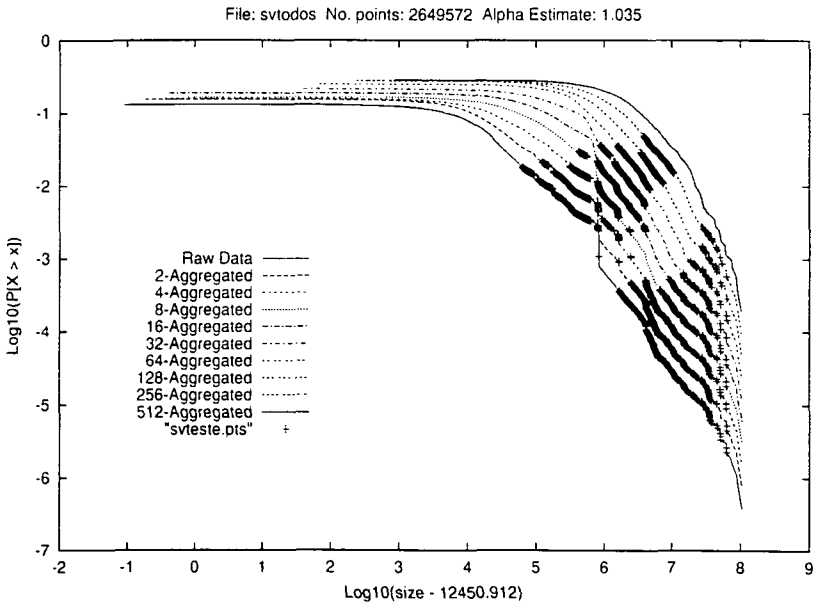


(a) Distribuição acumulada das requisições *misses*

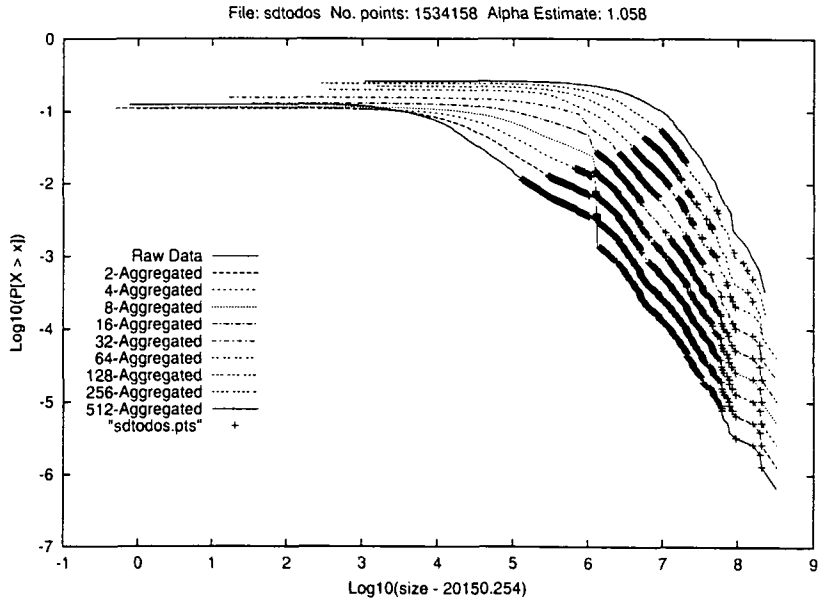


(b) Cauda das distribuições das requisições *misses*

Figura 5.6: Distribuição acumulada (a) e cauda das distribuições (b) dos tamanhos das requisições e dos tempos de serviço dos *misses*.

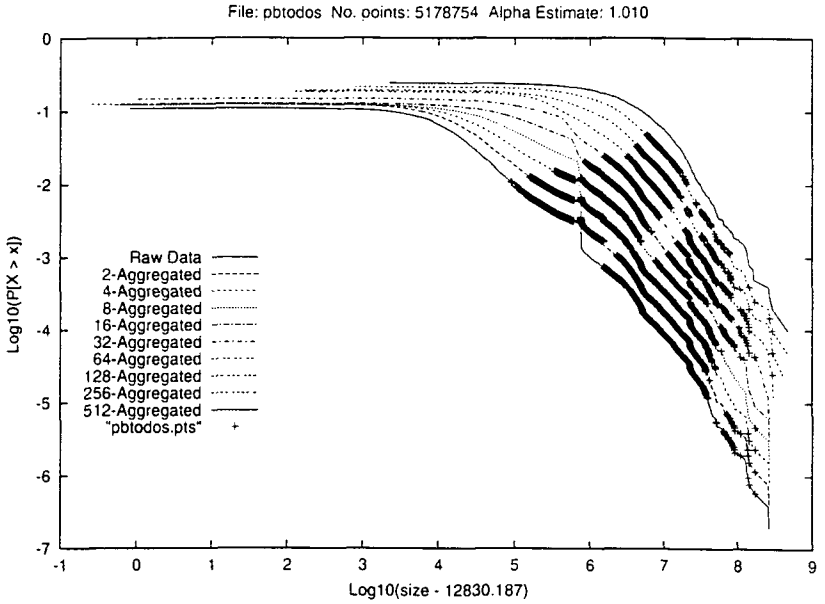


(a) cache SV

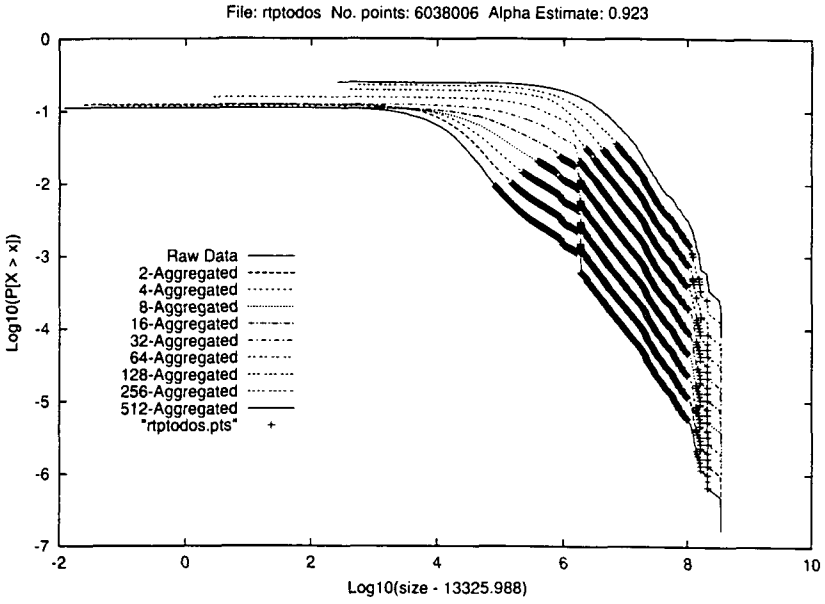


(b) cache SD

Figura 5.7: Região da cauda da distribuição de *Pareto* para os caches SV(a) e SD(b).



(a) cache PB



(b) cache RTP

Figura 5.8: Região da cauda da distribuição de *Pareto* para os caches PB(a) e RTP(b).

que é normalmente aplicada para modelar conjuntos de dados que se distinguem pela cauda. Optamos, então, pelas distribuições mais utilizadas para modelar o corpo do conjunto de dados descritas na literatura [1, 4, 5, 9, 15]. Testamos as distribuições *Weibull* e *Lognormal* para modelar os tempos de serviço e tamanhos das respostas. Os parâmetros necessários para a definição das distribuições foram calculados por um programa criado para esta finalidade. As entradas de dados deste programa são os conjuntos de dados a serem modelados.

Para obter os parâmetros da distribuição *Lognormal* foram calculados as médias e os desvios padrão do logaritmo natural dos tamanhos das respostas e dos tempos de serviço. As fórmulas da média (μ) e do desvio padrão (σ) usadas para o cálculo dos parâmetros são, respectivamente, apresentadas a seguir:

$$\mu = \frac{\sum_{i=1}^{n} ln(x_i)}{n}$$

$$\sigma = \frac{\sqrt{n\sum_{i=1}^{n} ln(x_i)^2 - (\sum_{i=1}^{n} ln(x_i))^2}}{n(n-1)}$$

A Tabela 5.1 apresenta os valores dos parâmetros da distribuição *Lognormal*, utilizados para modelar os conjuntos de tamanhos e tempos para cada cache. Podemos observar as diferenças nos modelos das duas variáveis. As médias obtidas apresentam diferenças significativas quando comparadas as médias do tamanho (variaram entre 7,90 e 8,22) e do tempo (variaram entre 5,12 e 5,82), sendo que, as médias dos tempos sempre foram menores. Os desvios padrão obtidos apresentam pequenas diferenças quando comparados com os desvios do tamanho (variaram entre 1,44 e 1,69) e do tempo (variaram entre 1,37 e 1,95). Os desvios padrão dos tempos observados, em sua maioria, são maiores dos que os observados para os tamanhos, a exceção é o cache PB. É importante ressaltar que os parâmetros utilizados na geração dos gráficos para avaliação da distribuição *Lognormal* não sofreram ajustes, ou seja, os parâmetros foram calculados obtendo-se a média e o desvio padrão do logaritmo natural dos tamanhos e tempos.

Como descrito no capítulo 2, a distribuição *Weibull* tem em sua fórmula dois parâmetros principais, onde α é o parâmetro de escala e β é o parâmetro de forma. Os dois parâmetros devem ser maiores do que zero. Para obter os parâmetros da distribuição *Weibull*, a função de distribuição cumulativa foi reduzida para a equação da reta $y = mx + b$, onde m indica a

Cache	Tamanho		Tempo	
	μ	σ	μ	σ
PB	7,90	1,56	5,81	1,37
RTP	8,09	1,44	5,12	1,51
SD	8,06	1,69	5,82	1,95
SV	8,22	1,44	5,57	1,52

Tabela 5.1: Parâmetros da distribuição *Lognormal*.

inclinação da reta e b o ponto de intersecção com o eixo y. A redução foi realizada da seguinte forma:

$$f(x) = 1 - e^{-(\frac{x}{\alpha})^{\beta}},$$

$$1 - f(x) = e^{-(\frac{x}{\alpha})^{\beta}},$$

$$ln(\frac{1}{1-(fx)}) = (\frac{x}{\alpha})^{\beta},$$

$$ln[ln(\frac{1}{1-(fx)})] = \beta ln(\frac{x}{\alpha}),$$

$$ln[ln(\frac{1}{1-(fx)})] = \beta lnx - \beta ln\alpha.$$

Comparando a equação com a fórmula reduzida, é possível verificar que o lado esquerdo da equação corresponde a y na fórmula reduzida, lnx corresponde a x, β corresponde a m, $-\beta\ ln\alpha$ corresponde a b. Então, realizando uma regressão linear, com os dados da distribuição cumulativa, os parâmetros β e b são obtidos diretamente. A estimativa do parâmetro α foi calculado como segue:

$$\alpha = e^{-}(\frac{b}{\beta}).$$

A Tabela 5.2 apresenta os valores dos parâmetros da distribuição *Weibull* utilizados para modelar os conjuntos de tamanhos e tempos para cada cache. Os valores obtidos para o parâmetro de escala (α) apresentam diferenças significativas quando comparados com os valores de α obtidos para modelar o tamanho (variaram entre 10.704 e 14.754) e do tempo (variaram entre

Cache	Tamanho		Tempo	
	α	β	α	β
PB	10.704	0,42	2.351	0.50
RTP	12.691	0,58	558	0,47
SD	11.106	0,51	2.209	0,41
SV	14.754	0,62	1.627	0,42

Tabela 5.2: Parâmetros da distribuição *Weibull*.

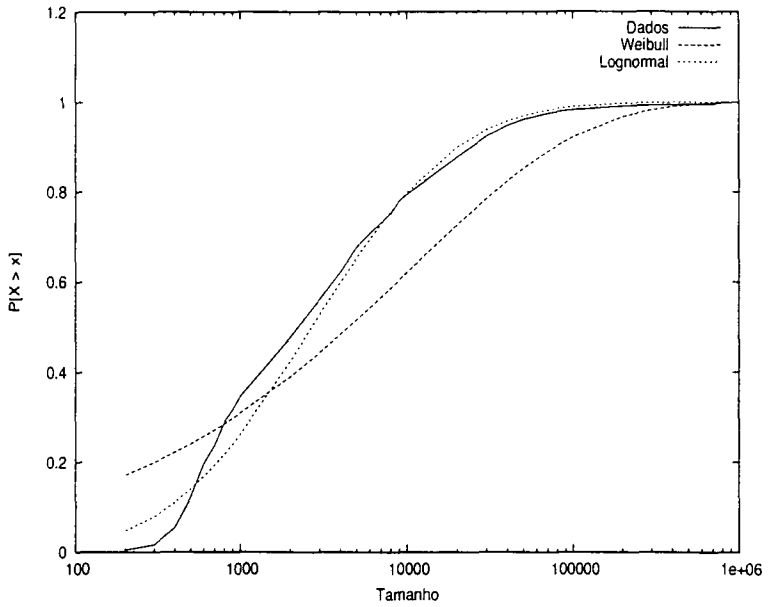
558 e 2.351), sendo que, os valores do parâmetro de escala dos tempos sempre foram menores. Os valores do parâmetro da forma (β) apresentam pequenas diferenças quando comparados os valores do tamanho (variaram entre 0,42 e 0,62) e do tempo (variaram entre 0,41 e 0,50). Os valores de (β), para os tamanhos, são, em sua maioria, maiores dos que os observados para os tempos, a exceção é o cache PB. Novamente é possível observar as diferenças nos modelos das duas variáveis, os parâmetros de escala tem grande variação enquanto os parâmetros de forma sofrem pequena variação. Os parâmetros utilizados na geração dos gráficos para avaliação da distribuição *Weibull* também não sofreram ajustes.

Os gráficos para os dados reais e os modelos gerados, para tamanhos das respostas e tempo de serviço dos caches, são apresentados nas Figuras 5.9, 5.10, 5.11 e 5.12. Os conjuntos de dados de todos os caches foram modelados de forma similar. Uma inspeção visual indica que o modelo *Lognormal* representa adequadamente os conjuntos de dados.

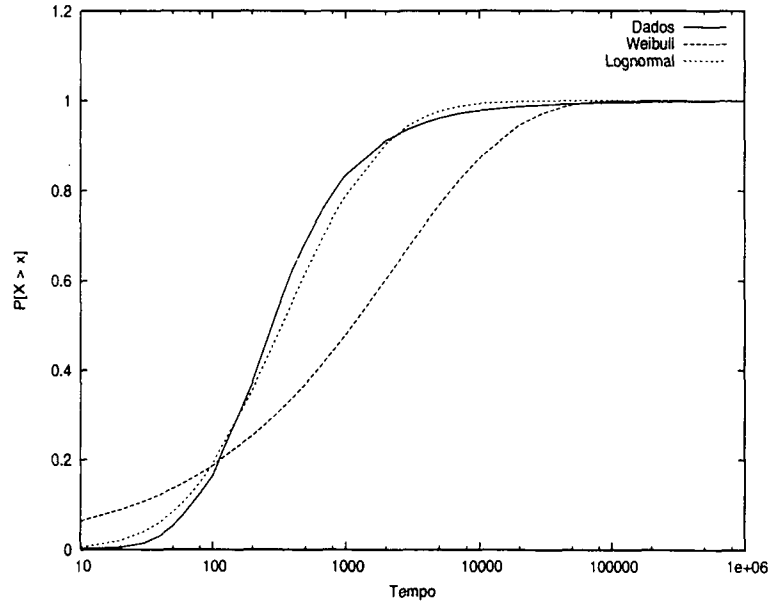
Para verificar a validade do modelo foram calculados os valores referentes ao Qui-quadrado (χ^2) das duas distribuições. Para obter o χ^2 das distribuições foi utilizada a seguinte fórmula:

$$\chi^2 = \sum \frac{(o-e)^2}{e},$$

sendo que o representa a freqüência real observada nos dados e e representa a freqüência esperada pela distribuição. Na Tabela 5.3 são apresentados os resultados obtidos referentes ao χ^2. Os resultados indicam que a distribuição *Lognormal* é mais adequada para o modelo de tamanho das respostas e de tempos de serviço do que a distribuição *Weibull*.

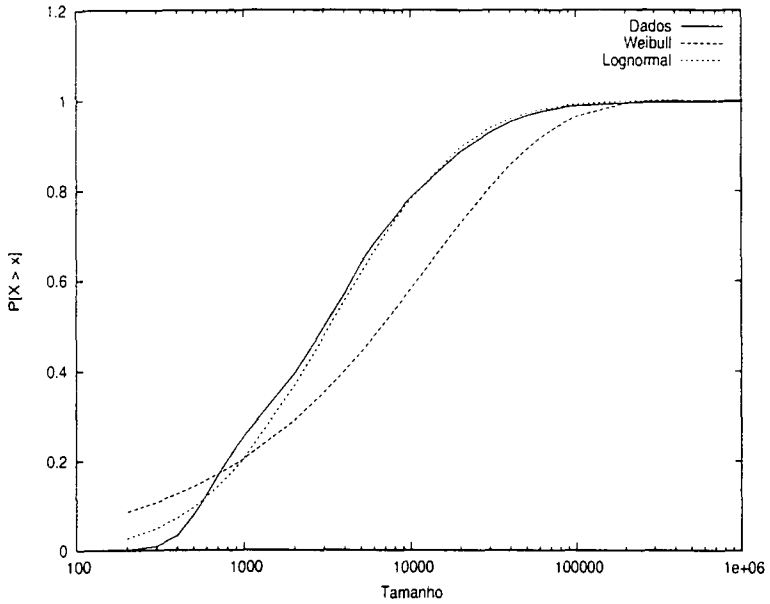


(a) Distribuição e modelos para os tamanhos

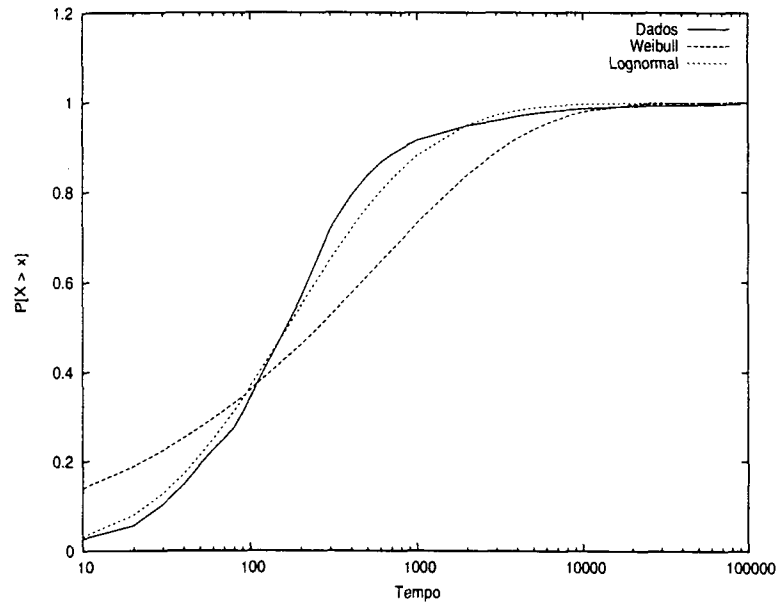


(b) Distribuição e modelos para os tempos

Figura 5.9: Distribuições empírica e modeladas para os tamanhos (a) e tempos (b). Dados do cache PB.

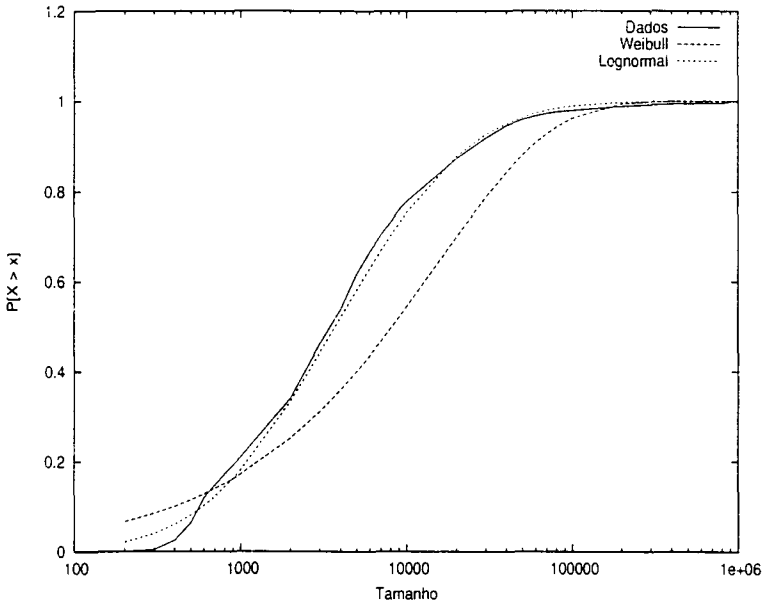


(a) Distribuição e modelos para os tamanhos

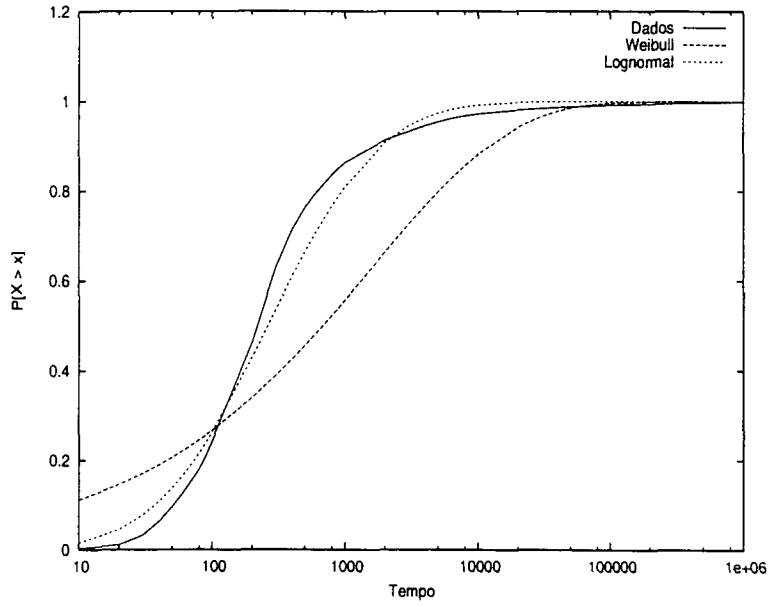


(b) Distribuição e modelos para os tempos

Figura 5.10: Distribuições empírica e modeladas para os tamanhos (a) e tempos (b). Dados do cache RTP.

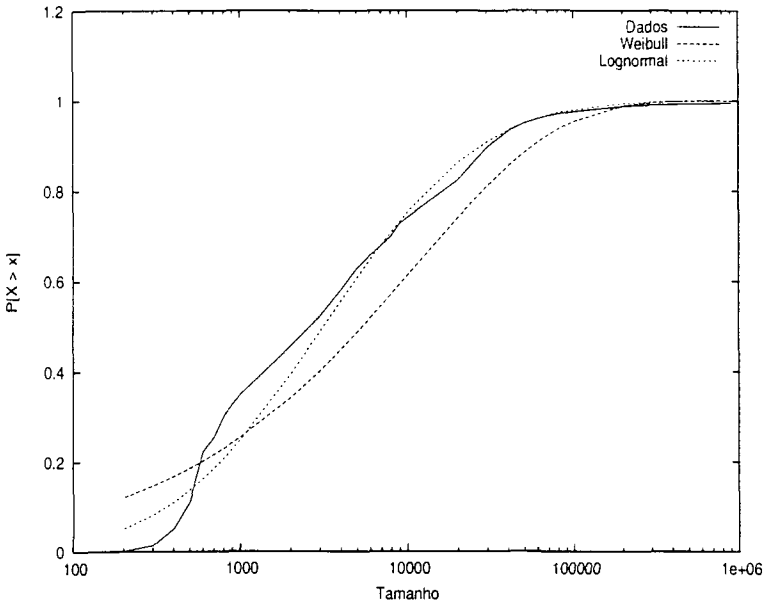


(a) Distribuição e modelos para os tamanhos

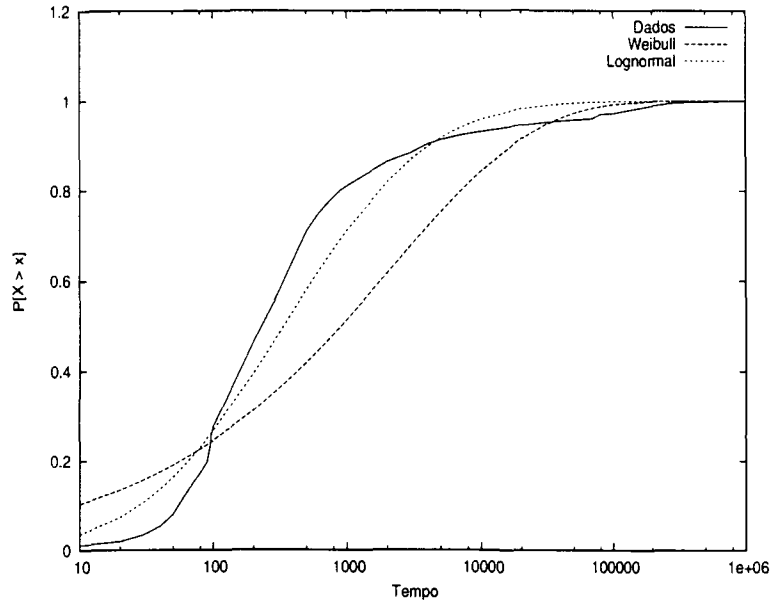


(b) Distribuição e modelos para os tempos

Figura 5.11: Distribuições empírica e modeladas para os tamanhos (a) e tempos (b). Dados do cache SV.



(a) Distribuição e modelos para os tamanhos



(b) Distribuição e modelos para os tempos

Figura 5.12: Distribuições empírica e modeladas para os tamanhos (a) e tempos (b). Dados do cache SD.

Cache	Tamanho		Tempo	
	Weibull	*Lognormal*	*Weibull*	*Lognormal*
PB	8.72	0,74	5.96	0,20
RTP	10,41	1,01	1.54	0,07
SD	14.53	2,73	3,60	0,83
SV	8.60	0,94	8,07	0,38

Tabela 5.3: Valores do χ^2 para os modelos testados.

5.3 Considerações Finais

Neste capítulo apresentamos gráficos para as freqüências simples e acumulada dos tamanhos das respostas e dos tempos de serviço. Propusemos um modelo para os tempos de serviço. É possível, pela inspeção visual e dos dados obtidos através do χ^2, perceber que os modelos de distribuição *Lognormal* representam adequadamente os conjuntos de dados do tamanho da resposta e do tempo de serviço.

No próximo capítulo são apresentadas as taxas de serviço nos servidores. São feitas avaliações sobre o comportamento dos *hits* e dos *misses*, para identificar suas influências nas taxas de serviço efetivas.

Capítulo 6

Taxas de Serviço nos Servidores Cache

Neste capítulo são apresentadas as taxas de serviço (*throughput*) nos servidores, obtidas a partir dos dados coletados. Para realizar estes cálculos foram necessários filtros adicionais aos já descritos no capítulo 3. Para o cálculo da taxa de serviço foram analisadas apenas as requisições cujo código de resposta era 200, pois representam a transmissão completa de um arquivo, estando ele ou não no cache. Também foram aplicados filtros para identificar apenas as respostas do código 304, pois representam que o arquivo que se encontra no cache ainda é válido. Análises sobre este tipo de resposta são apresentadas para avaliar o comportamento do tempo de resposta em conjuntos de respostas cujo tamanho apresenta pequena variabilidade.

6.1 Avaliação das Taxas de Serviço em Grupos Distintos

A taxa de serviço foi calculada dividindo-se as requisições em grupos de *hits* e *misses* e de acordo com as classes de tamanhos já descritas anteriormente na seção 4.2. Para se obter as taxas, os tamanhos foram convertidos para Kbytes e os tempos convertidos em segundos. Para cada arquivo servido foi obtida a taxa de serviço dividindo-se seu tamanho pelo tempo de serviço. A seguir a taxa de serviço média, para cada conjunto, foi calculada.

Na Tabela 6.1 são apresentados os dados referentes a taxa de serviço calculada. Na primeira

coluna são mostrados os caches, na segunda coluna o tipo de resposta (*hit* ou *miss*) e nas colunas subseqüentes as classes de tamanhos já descritas anteriormente.

Cache	tipo	Taxa de Serviço por Classe em Kbytes/s				
		Classe 1	Classe 2	Classe 3	Classe 4	Todos
SD	*hit*	52,41	150,23	145,27	187,36	57,36
	miss	5,85	37,95	34,02	21,09	20,06
RTP	*hit*	101,48	408,46	178,50	109,41	108,34
	miss	10,39	58,90	71,98	56,62	35,13
SV	*hit*	19,22	93,57	70,07	98,44	29,25
	miss	7,14	43,87	39,42	18,16	26,24
PB	*hit*	19,50	85,94	89,51	67,68	28,97
	miss	4,32	29,26	49,54	35,66	13,76

Tabela 6.1: Taxa de serviço por classes de tamanho para respostas com código 200.

É possível perceber a grande variabilidade de taxas obtidas. As taxas são sempre menores na classe 1, que compreende os arquivos menores. Isto se deve à proporção do tempo necessário para estabelecimento das conexões de rede em relação ao tempo total da transmissão. As transferências desta classe são realizadas com um pequeno número de pacotes. As conexões necessárias no tratamento dos pacotes TCP influenciam significativamente o desempenho de transmissão de arquivos que se enquadram nesta classe de tamanho. As classes que têm arquivos de tamanhos maiores apresentam melhor desempenho.

As classes 2, 3 e 4 apresentam taxa de serviço similar. Nestas classes o estabelecimento da conexão não influencia tanto a obtenção dos tempos de serviço, tornando as taxas mais próximas, diferentemente do observado na classe 1. É possível identificar claramente que as respostas *hits* tem desempenho de duas a oito vezes melhor que as respostas *misses* em todos os caches avaliados e para todas as classes. As respostas *hits* tem melhor desempenho devido ao recurso solicitado estar mais próximo do cliente. Assim, o número de *hosts* que uma requisição *hit* precisa atravessar é menor do que o número de *hosts* visitados nas requisições *misses*, como conseqüência as requisições *misses* tem desempenho menos eficiente. Outra justificativa para

estas diferenças entre *hits* e *misses* é pela melhor conectividade entre os clientes e os caches. Os caches analisados pertencem à Internet 2 americana. As maiores diferenças são observadas na classe 1 em que a taxa de serviço para os *hits* chega a ser dez vezes superior à dos *misses*. Os arquivos maiores têm melhor desempenho pois o custo inicial de abertura de conexão é minimizado com a transmissão de um número maior de pacotes para completar a requisição.

Comparando com os dados disponíveis na literatura verifica-se que as requisições *misses* obtiveram desempenho bem superior. Anteriormente, as taxas de serviço para estas requisições chegavam no máximo a 7 Kbits/s, ou seja, menos do que 1 Kbyte/s [34]. No presente trabalho as taxas obtidas para as requisições *misses* variaram entre 13,76 e 35,13 Kbytes/s considerando todas as requisições. Portanto, a melhora nas taxas observada foi de 13 a 35 vezes. Esta melhora nos tempos dos *misses* se deve a melhora na conectividade dos clientes e servidores. Também existem esforços dos pesquisadores para minimizar as diferenças entre as respostas das requisições *hits* e *misses* através da redução de velocidade dos *hits* [19]. As requisições *hits* também tiveram o seu desempenho melhorado. Comparando com os dados de caches similares avaliados anteriormente, esta melhora chega a quatro vezes.

A velocidade de serviço na Internet está associada aos recursos básicos de comunicação associados à Internet. Entre os recursos estão as capacidades de transmissão dos dados, as quais são bem diversas e tem velocidades que variam entre 56 Kbits/s e 1 Gbit/s [3]. Estas diferentes tecnologias influenciam diretamente as taxas de serviço obtidas pelos usuários. As taxas de serviço obtidas neste trabalho variaram entre 4,32 Kbytes/s e 408,46 Kbytes/s, ou seja, as taxas de serviço estão muito distantes da capacidade de transmissão de dados disponível demonstrando que os outros recursos envolvidos no tratamento de uma requisição têm influência significativa para as taxas de serviço.

A Tabela 6.2 apresenta os mesmos dados da Tabela 6.1 na unidade pacotes por segundo. O tamanho dos pacotes foi definido em 1.500 bytes, pois é um dos mais comuns nas transmissões Web [13]. Os tamanhos de pacotes que trafegam pela Web são variáveis. O objetivo deste processamento foi verificar as características das transmissões em pacotes e obter a taxa de pacotes transmitidos por segundo.

Na Tabela 6.3 são mostradas as taxas obtidas nas requisições em que as respostas foram

Cache	tipo	pacotes por segundo				
		Classe 1	Classe 2	Classe 3	Classe 4	Todos
SD	*hit*	56,30	114,71	101,20	128,04	61,53
	miss	8,80	28,31	23,61	14,41	17,54
RTP	*hit*	123,99	316,09	123,95	74,72	149,40
	miss	13,42	44,56	49,92	38,68	30,37
SV	*hit*	24,63	71,28	48,64	67,24	36,27
	miss	9,20	33,24	27,36	12,41	21,37
PB	*hit*	24,81	65,58	62,13	46,23	33,84
	miss	6,10	22,03	34,33	24,37	12,86

Tabela 6.2: Pacotes transmitidos por classes de tamanho e para respostas com código 200 divididos em pacotes de 1500 bytes.

com o código 304 para todos os caches. Este tipo de resposta é enviada quando o navegador ou cache no nível inferior contém a página requisitada pelo cliente mas esta página está com prazo de validade vencido (*stalled*). Neste caso, o cache que contém a página emite uma requisição condicional para verificar se a página foi modificada, o servidor informa que a página não sofreu alterações e continua sendo válida. Não há divisão em classes de tamanhos, pois todos os registros do código de resposta 304 se enquadram na classe 1. Como a variação dos tamanhos para estas respostas é muito pequena, é possível fazer uma avaliação da eficiência de cada um dos caches. Os caches avaliados atendem conexões de diferentes continentes e o cache RTP apenas conexões localizadas nos Estados Unidos.

Os resultados não apresentam a mesma variação das outras respostas, devido à similaridade dos tamanhos transferidos. Em todos os casos é necessária a transmissão de um único pacote para transferência da requisição pois o tamanho médio das solicitações varia muito pouco (entre 270 e 280 bytes). O cache RTP teve uma taxa de 34,71 Kbytes/s, sendo este valor significativamente maior que os apresentados pelos demais caches. Neste cache, 98% de todas as requisições foram atendidas em um tempo menor do que 100 milissegundos. Este desempenho melhor se deve a proximidade do cache aos clientes e as conexões de alta velocidade presentes no país onde ele

Cache	tipo	Taxa de serviço em Kbytes/s
SD	hit	12,49
	miss	2,93
RTP	hit	34,71
	miss	5,68
SV	hit	7,46
	miss	2,62
PB	hit	6,26
	miss	2,10

Tabela 6.3: Taxa de serviço por classes para respostas com código 304.

está sediado.

Para obtenção dos dados apresentados na Tabela 6.4 foram feitos cálculos similares ao da Tabela 6.3, considerando a taxa obtida em pacotes por segundos. Comparando com os dados da Tabela 6.3, é possível perceber que a taxa pacotes/segundo é maior que a taxa Kbytes/s. Isto se deve ao fato de que as respostas são pequenas e não preenchem um pacote completo, isto aumenta a taxa pacotes/segundo.

6.2 Efetividade do Cache

Podemos definir a efetividade dos caches como a taxa média de resposta de todas as requisições *hits* para um conjunto de registros dividida pela taxa média de resposta das requisições *misses* do mesmo conjunto de registros. A efetividade indica o ganho percebido por um usuário quando a página é recuperada de um cache. A efetividade foi calculada e os resultados obtidos são apresentados na Tabela 6.5. É possível perceber que o cache aumenta a efetividade do serviço HTTP. Um *hit* é atendido com uma taxa média de serviço de 3 a 6 vezes maior do que a taxa dos *misses*.

Cache	tipo	pacotes por segundo
SD	hit	47.12
	miss	9,98
RTP	hit	124,44
	miss	20,78
SV	hit	26,75
	miss	9.42
PB	hit	23,35
	miss	7,92

Tabela 6.4: Taxa de serviço em pacotes por segundo para respostas 304.

Cache	efetividade do *proxy*
SD	4,26
RTP	6,11
SV	2,84
PB	2,98

Tabela 6.5: Dados da efetividade do cache.

6.3 Considerações Finais

Neste capítulo foram apresentadas as taxas de serviço obtidas nos servidores cache, e as avaliações mais relevantes são:

- Os arquivos pequenos (classe 1) experimentam uma taxa de serviço menor, e as respostas *hits* chegam a ser dez vezes mais rápidas do que as respostas *misses*. Isto se deve à proporção do tempo necessário para estabelecimento das conexões de rede em relação ao tempo total da transmissão. Como o número de pacotes transmitidos nos arquivos pequenos é menor, a sobrecarga da abertura de conexões influencia fortemente a taxa de serviço. As respostas *misses* pequenas têm desempenho bastante reduzido, pois há necessidade de estabelecimento de conexões complementares com os servidores o que afeta

ainda mais a taxa de serviço.

- As classes 2, 3 e 4 apresentam taxas de serviço que chegam a 408,46 Kbytes/s nas requisições *hits* e a 71,98 Kbytes/s nas requisições *misses*, demonstrando que, tanto as requisições *hits* quanto as *misses* têm tido um melhor desempenho com o passar dos anos;

- O cache RTP têm o melhor desempenho, isto pode ser observado pelas taxas de serviço obtidas para as requisições *hits*. Como o cache RTP responde a requisições localizadas no país em que está localizado, possivelmente atravessa um número de *hosts* menor e quando o número de *hosts* no caminho é menor, o tempo de serviço é melhor. Este cache também é o mais efetivo, ou seja, onde as respostas *hits* são enviadas com taxas muito superiores as taxas das respostas *misses*;

- O custo para as requisições *misses* varia entre 2,84 e 6,11 quando comparada com as requisições *hits*. É possível verificar que houve uma melhora na conectividade entre clientes e servidores.

Capítulo 7

Conclusão

Dissertação apresentou uma descrição da carga dos caches Web, bem como uma comparação com os resultados disponíveis na literatura, com o intuito de adquirir uma melhor percepção e entendimento das cargas de servidores *proxy*, assim como retratar os padrões de acesso. O objetivo principal da descrição da carga é apresentar uma avaliação da evolução das características da carga Web em termos de variabilidade, bem como os reflexos desta variabilidade nas taxas de serviço.

Também são apresentadas evidências de que os tamanhos das respostas apresentam comportamentos distintos dos tempos de serviço. Através da análise de uma série de gráficos percebeu-se que, embora ambas as características possam ser modeladas pela mesma distribuição, os parâmetros são distintos, em especial a média. Estas diferenças apresentadas são importantes pois os modelos de simulação e analíticos precisam dos modelos de tempo de serviço.

Este capítulo conclui esta dissertação apresentando os resultados obtidos, as contribuições geradas e sugestões para trabalhos futuros.

7.1 Resultados e Contribuições

As contribuições deste trabalho podem ser divididas em dois grupos: estudo de caracterização de carga Web comparando com os resultados disponíveis na bibliografia e apresentação de modelos estatísticos com parâmetros distintos para os tamanhos das respostas e para os tempos de serviço.

O primeiro conjunto de contribuições deste trabalho é a caracterização da carga de quatro servidores cache, tomada no período de 28/10/2002 a 31/10/2002, totalizando 18.630.327 requisições. Juntamente com esta caracterização foi apresentada uma série histórica coletada em trabalhos anteriores de caracterização de carga e uma comparação com os dados gerados por este trabalho. As características mais relevantes verificadas e comparadas são:

- O tamanho médio dos arquivos variou entre 12,16 e 19,68 Kbytes e a mediana variou entre 0,7 e 1,5 Kbytes. Os maiores arquivos encontrados nos registros estão entre 156 e 377 Mbytes. Observamos que, no período de 8 anos, o tamanho médio teve um pequeno aumento, a mediana dos arquivos diminuiu, e o tamanho dos maiores arquivos teve crescimento acentuado. Estes dados mostram que mesmo com grandes arquivos sendo transmitidos, isso ainda não causou impacto nos tamanhos médios dos arquivos transferidos devido ao grande volume no tráfego de arquivos pequenos;

- As requisições de imagens (*gif e jpeg*) representam uma fração significativa, em torno de 48% das requisições e 21% da fração de bytes, e o tamanho médio das imagens ficou em torno de 6 Kbytes. Comparando com trabalhos anteriores é possível observar que a fração das requisições, o tamanho médio e a fração dos bytes das imagens têm diminuido com o passar dos anos. Nos trabalhos anteriores, o tamanho médio das imagens era de 8 Kbytes, a fração das requisições era de 68% e a fração dos bytes era de 40%. Os arquivos do tipo HTML continuam tendo um grande número de requisições, em torno de 19%. A fração de arquivos HTML se mantém praticamente inalterada;

- Em torno de 7% das requisições tem tamanho maior do que 70 Kbytes. Estes arquivos são responsáveis por 60% dos bytes transferidos;

- Os coeficientes de variação para *hits* estão entre 8,65 e 22,72 para os tamanhos dos arquivos servidos pelos quatro caches e entre 9,21 e 22,35 para os tempos de serviço. Comparando com dados obtidos na literatura é possível verificar que o coeficiente de variação para o tempo diminuiu, estava entre 19,54 e 58,53, e que coeficiente de variação para os tamanhos aumentou, estava entre 5,78 e 11,89. Uma possível justificativa para estas mudanças é que os tamanhos têm coeficiente de variação maiores devido ao crescimento dos arquivos

grandes e principalmente pela popularização destes arquivos. Os arquivos de áudio, vídeo e de programas compactados já podem ser obtidos em um tempo razoável pelos usuários da Web, pois as conexões estão cada vez mais velozes. A possível justificativa para a diminuição do coeficiente de variação dos tempos é a evolução das capacidades das redes e dos sistemas servidores disponíveis para os usuários. Os coeficientes de variação para os *misses* variaram entre 19,68 e 37,26 para os tamanhos e entre 10,28 e 28,20 para os tempos. Comparando com os dados publicados anteriormente, percebe-se que os coeficientes de variação dos *misses* aumentaram. Anteriormente os coeficientes de variação para os *misses* variavam entre 7,75 e 42,35 para os tamanhos e entre 5,46 e 13,53 para os tempos. Uma possível justificativa para essa variação nos tamanhos é o fato de que os arquivos maiores não são constantemente armazenados nos caches e geralmente necessitam serem solicitados dos servidores, gerando um maior coeficiente de variação. Para os tempos, a justificativa é o fato de que o caminho a ser seguido pela requisição atravessa um número maior de *hosts*, quanto maior o número de *hosts* maior a probabilidade de atrasos e de aumento do coeficiente de variação. Estes dados mostram que a variabilidade dos tamanhos dos arquivos aumentou, como já se previa;

- Em arquivos com mais de 30 Kbytes existe forte correlação entre o tamanho da resposta e o tempo de serviço. Em arquivos menores a correlação é pequena. O principal motivo para a fraca correlação nos arquivos pequenos é o custo inicial do protocolo para cada conexão. Este custo é amortizado em arquivos que precisem transmitir um número maior de pacotes para completar a requisição;

- Os arquivos pequenos experimentam uma taxa de serviço menor, e as respostas *hits* chegam a ser dez vezes mais rápidas do que as respostas *misses*. Isto se deve à proporção do tempo necessário para estabelecimento das conexões de rede em relação ao tempo total da transmissão. Como o número de pacotes transmitidos nos arquivos pequenos é menor, o estabelecimento de conexões influencia fortemente a taxa de serviço. As respostas *misses* pequenas têm desempenho bastante reduzido, há necessidade de estabelecimento de conexões complementares com o servidores o que afeta ainda mais a taxa de serviço.

Como segunda contribuição desta dissertação, apresentamos uma proposta de um modelo de serviços HTTP. Os tempos de serviço foram modelados pela distribuição *lognormal*. Como verificado nos dados produzidos pela caracterização da carga, os tempos de serviço podem ser diferentes em várias ordens de grandeza para arquivos de mesmo tamanho. Assim, utilizar parâmetros dos modelos de tamanho da resposta em substituição aos modelos de tempo de serviço pode levar a conclusões errôneas. Acreditamos que o cache pode se beneficiar se reconhecer que requisições do mesmo arquivo são atendidas por conexões diferentes, que apresentam qualidade diferente de conectividade.

7.2 Trabalhos Futuros

A caracterização continuada é necessária, pois a evolução da Web é constante. O número de usuários e volume de documentos armazenados crescem constantemente e a mudança nas cargas devido a novas aplicações e novas facilidades de uso é freqüente. Um estudo da evolução dos acessos a documentos únicos seria útil para avaliar a presença destes documentos nos caches e gerar informações que possibilitem um tratamento mais adequado destes registros pelo cache.

Mesmo com o grande volume de registros de acesso disponíveis, existem poucos trabalhos de caracterização de tempos de serviço de caches publicados, dificultando a obtenção de dados estatísticos e também o entendimento das razões de longos tempos de resposta observados freqüentemente no atendimento das requisições. Uma seqüência natural deste trabalho é a utilização dos modelos propostos em modelos analíticos e de simulação para caches e servidores Web.

Referências Bibliográficas

[1] Ghaleb Abdulla. *Analysis and Modeling of World Wide Web Traffic.* PhD thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, May 1998.

[2] Ghaleb Abdulla, Edward A. Fox, Marc Abrams, and Stephen Williams. WWW Proxy Traffic Characterization with Application to Caching. *Computer Science Department, Virginia Technology*, (CS-97-03):1–20, March 1998.

[3] Abilene. http://abilene.internet2.edu.

[4] Martin F. Arlitt and Tai Jin. A Workload Characterization Study of the 1998 World Cup Web Site. *IEEE Network*, 14(3):30–37, May/June 2000.

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

[6] David M. Kristol Balachander Krishnamurthy, Jeffrey C. Mogul. Key Differences Between HTTP/1.0 and HTTP/1.1. volume 31, pages 1737–1751, Toronto, 1999.

[7] Gaurav Banga and Peter Druschel. Measuring the Capacity of a Web Server Under Realistic Loads . *World Wide Web Journal - Special Issue on World Wide Web Characterization and Performance Evaluation*, 2(1-2):69–83, 1999.

[8] Nikhil Bansal and Mor Harchol-Balter. Analysis of SRPT Scheduling: Investigating Unfairness. In *Proceedings of ACM Sigmetrics 2001 Conference on Measurement and Modeling of Computer Systems*, pages 279–290, 2001.

[9] Paul Barford and Mark Crovella. Generating Representative Web Workloads for Network and Server Performance Evaluation. In *Proceedings of ACM SIGMETRICS'98*, pages 151–160, Madison, Wisconsin, USA, July 1998.

[10] Paul Barford and Mark Crovella. Measuring Web Performance in the Wide Area. *Performance Evaluation Review - Special Issue on Network Traffic Measurement and Workload Characterization*, 27(2):35–46, September 1999.

[11] Greg Barish and Katia Obraczka. World Wide Web Caching: Trends and Techniques. *IEEE Communications Magazine*, 38(5):178–184, 2000.

[12] Joachim Charzinski. HTTP/TCP Connection and Flow Characteristics. *Performance Evaluation*, 42:149–162, September 2000.

[13] K Claffy, Greg Miller, and Kevin Thompson. The Nature of the Beast: Recent Traffic Measurements from an Internet Backbone. In *Proceedings of the INET98*, Geneva, 1998.

[14] Douglas E. Comer and David L. Stevens. *Interligação em Rede com TCP/IP: Projeto, Implementação e Detalhes Internos.* Editora Campus, Rio de Janeiro, 1999.

[15] Mark Crovella and Azer Bestavros. Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes. *IEEE/ACM Transaction Networking*, 5:835–846, December 1997.

[16] Mark Crovella, Bob Frangioso, and Mor Harchol-Balter. Connection Scheduling in Web Servers. In *Proceedings of USITS'99: USENIX Symposium on Internet Technologies and Systems*, pages 243–254, Boulder, Colorado, USA, October 1999.

[17] Mark E. Crovella and Murad S. Taqqu. Estimating the Heavy Tail Index from Scaling Properties. *Methodology and Computing in Applied Probability*, 1(1):55–79, July 1999.

[18] Brian D. Davison. A Web Caching Primer. *IEEE Internet Computing*, 5(4):38–45, July/August 2001.

[19] Brian D. Davison, Chandrasekar Krishnan, and Baoning Wu. When does a hit = a miss? In *Proceedings of the Seventh International Workshop on Web Content Caching and Distribution (WCW'02)*, Boulder, CO, August 2002.

[20] John Dilley, Rich Friedrich, Tai Jin, and Jerome Rolia. Measurement Tools and Modeling Techniques for Evaluating Web Server Performance. In *Proceedings of 9th International Conference on Modeling Techniques and Tools*, pages 155–168, Lecture Notes in Computer Science, vol. 1245, Springer-Verlag, 1997.

[21] James Gettys, Jeffrey C. Mogul, Henrik Frystyk, Paul J. Leach, Larry Masinter, and Tim Berners-Lee. Hypertext Transfer Protocol - HTTP/1.1. Technical report, 1999.

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

[23] Venkata N. Padmanabhan Hari Balakrishnan, Srinivasan Senhan, Mark Stemm, and Randy H. Katz. TCP Behavior of a Busy Internet Server: Analysis and Improvements. In *Proceedings of the IEEE Infocom 1998 Conference*, pages 252–262, San Francisco, CA, April 1998.

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

[25] Wagner Meira Jr., Cristina Duarte Murta, Sérgio Vale Aguiar Campos, and Dorgival Olavo Guedes Neto. *Sistemas de Comércio Eletrônico.* Editora Campus, Rio de Janeiro, 2002.

[26] Balachander Krishnamurthy and Jennifer Rexford. *Web Protocols and Practice: HTTP/1.1, Networking Protocols, Caching, and Traffic Measurement.* Addison-Wesley Longman Publishing Co., Inc., 2001.

[27] Balachander Krishnamurthy and Craig E. Wills. Analyzing Factors That Influence End-to-End Web Performance. In *Proceedings of the 9th International World Wide Web Conference on Computer Networks*, volume 33, pages 17–32, Amsterdam, May 2000.

[28] Binzhang Liu. Characterizing Web Response Time. Master's thesis, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, April 1998.

[29] Ari Luotonen. *Web Proxy Servers.* Prentice-Hall, 1998.

[30] Anirban Mahanti, Carey Williamson, and Derek Eager. Traffic Analysis of a Web Proxy Hierarchy. *IEEE Network Magazine*, 14(3):16–23, May/June 2000.

[31] Daniel A. Menascé and Virgilio A. F. Almeida. *Capacity Planning for Web Performance. Metrics, Models, & Methods.* Prentice-Hall, New Jersey, 1998.

[32] Martin W. Murhammer, Orcun Atakan, Stefan Bretz, Larry R. Pugh, Kazunari Susuki, and David H. Wood. *TCP/IP: Tutorial e técnico.* Makron Books, São Paulo, 2000.

[33] Cristina Duarte Murta. *Modelo de Particionamento de Espaço para Caches na World Wide Web.* PhD thesis, Departamento de Ciência da Computação. Universidade Federal de Minas Gerais, Belo Horizonte, Minas Gerais, Agosto 1999.

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

[35] Cristina Duarte Murta and Tarcísio Paulo Corlassoli. Política de Escalonamento para Servidores Web Baseada na Velocidade da Conexão. *Anais do 20o. Simpósio Brasileiro de Redes de Computadores*, 2002.

[36] Cristina Duarte Murta and Marco Antonio Jonack. Caracterização de Carga de Caches na WWW. *Revista Eletrônica de Iniciação Científica*, 2(2), 2002.

[37] NLANR. National Laboratory for Applied Network Research. http://ircache.nlanr.net.

[38] David A. Patterson and John L. Hennessy. *Computer Architecture: A Quantitative Approach, 2nd Edition.* Morgan Kaufmann Publishers, California, 1996.

[39] David A. Patterson and John L. Hennessy. *Organização e Projeto de Computadores: A Interface Hardware/Software.* LTC, Rio de Janeiro, 2000.

[40] Alex Rousskov and Valery Soloviev. A Performance Study of the Squid Proxy on HTTP/1.0. *World Wide Web*, 2(1-2):47–67, 1999.

[41] Alex Rousskov and Duane Wessels. Cache digests. *Computer Networks and ISDN Systems*, 30(22-23):2155–2168, November 1998.

[42] Manish Sharma and John W. Byers. How Well Does File Size Predict Wide-Area Transfer Time? In *The Proceedings of Globecom 2002 - Global Internet Symposium 02*, Taipei, November 2002.

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

[44] William Stallings. *Arquitetura e Organização de Computadores.* Prentice Hall, São Paulo,

[45] Andrew S. Tanenbaum. *Computer Networks.* Prentice Hall. 1997.

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

[47] Gabriel Torres. *Hardware Curso completo.* Axcel Books do Brasil Editora, Rio de Janeiro, 1999.

[48] Duane Wessels and K Claffy. Evolution of the NLANR Cache Hierarchy: Global Configuration Challenges. *NLANR*, 1996.

Apêndice A

O *Software Squid*

Um dos programas para cache da Web mais utilizados é o *Squid*, que é o resultado do trabalho da comunidade da Internet, através do NLANR. É um programa de cache utilizado em servidores *proxy* para requisições de clientes Web. Implementa os protocolos FTP, HTTP, Gopher e WAIS (*Wide Area Information Servers*). A Figura A.1 mostra um diagrama de blocos com os protocolos suportados pelo *Squid* (servidor *proxy*). O *Squid* possui recursos para geração de *logs* das requições atendidas e é compatível com o SSL (*Secure Socket Layer*).

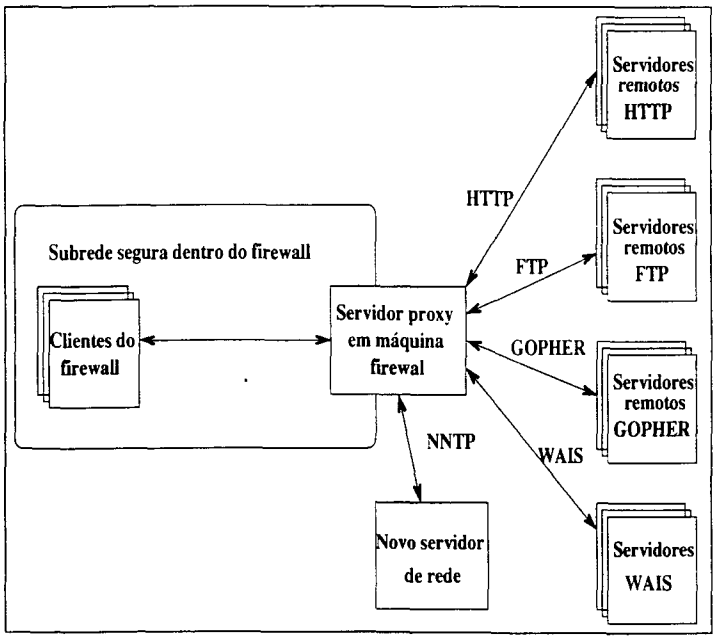


Figura A.1: Objetos suportados pelo *Squid*.

O *Squid* é um dos servidores de cache mais utilizados na Internet, estando presente na maioria dos servidores de cache [43]. Os principais motivos para isso são a sua simplicidade de configuração e utilização, e por ser gratuíto.

É possível utilizar servidores *Squid* organizados de forma hierárquica. Para executar a comunicação entre caches da mesma hierarquia, o *Squid* utiliza o protocolo ICP. A comunicação com outros caches é feita pela porta UDP 3130 por *default*, mas a comunicação também pode ser feita por HTTP. O *Squid* usa ACLs (*Access Control Lists*) para decidir quais clientes podem ser capazes de acessá-lo como um *proxy*.

Um Cache *Squid* tem a capacidade de funcionar como acelerador do servidor da Web. Ele ar na frente de um ou mais servidores origem e atuar como *front-end* para os pedidos recebidos. O servidor *Squid* deve ser executado na porta HTTP *default* 80 para funcionar como um servidor de *front-end*. Como muitos outros *proxies* com caching, o *Squid* utilizava originalmente um modelo de tempo de vida de expiração dos recursos entrados no cache. As versões recentes do *Squid* usam um modo diferente, baseado na taxa de atualização [26].

O *Squid* recentemente introduziu também os resumos de cache (*digests*). A versão atual do *Squid* (versão 2.5) é capaz de tratar de alguns dos recursos do HTTP 1.1, como conexões persistentes e autenticação de *proxy*. O *Squid* também é capaz de aproveitar as melhorias recentes nos sistemas operacionais populares, como as bibliotecas de *thread* [21]. O *Squid* foi concebido para ser um servidor *proxy* eficiente, com o menor tempo de resposta possível para o usuário. Para isso, mantém em memória RAM os objetos em trânsito, os objetos mais recentemente acessados, uma tabela com dados de todos os objetos armazenados no cache, as últimas chamadas de resolução de nomes e os últimos objetos acessados com erros. A manutenção dessas estruturas em memória principal tem objetivo de otimizar o funcionamento do *Squid*, evitando repetições de chamadas.

A.1 Códigos de Resultados do *Squid*

Os códigos de resultados informam como as requisições foram tratadas, se houve acerto, erro, dentre outros. Os códigos TCP referem-se às requisições HTTP (geralmente porta 3128). Os códigos UDP referem-se às requisições ICP (geralmente porta 3130). Os códigos *Squid* TCP e

UDP são os seguintes:

- TCP_HIT: uma cópia válida foi encontrada no cache.
- TCP_MISS: o objeto solicitado não se encontra no cache.
- TCP_REFRESH_HIT: uma cópia do objeto requisitado já expirada encontra-se no cache. O *Squid* fez um pedido para saber se o objeto foi modificado desde determinada data e recebeu a informação de que não houve alterações.
- TCP_REF_FAIL_HIT: uma cópia do objeto requisitado já expirada encontra-se no cache. *Squid* fez um pedido se modificado desde? Este pedido falhou (*timeout*) e por isso foi entregue essa cópia ao cliente.
- TCP_REFRESH_MISS: uma cópia do objeto requisitado já expirada encontra-se no cache. O *Squid* fez um pedido se modificado desde? E recebeu um novo objeto.
- TCP_CLIENT_REFRESH_MISS e TCP_CLIENT_REFRESH: o cliente emitiu um pedido para atualizar o objeto.
- TCP_IMS_HIT: o cliente emitiu um pedido se modificado desde? Uma cópia válida foi encontrada no cache.
- TCP_SWAPFAIL_MISS e TCP_SWAPFAIL: o objeto solicitado estava no cache mas não foi possível acessá-lo.
- TCP_NEGATIVE_HIT: o objeto solicitado não é armazenado no cache.
- TCP_MEM_HIT: uma cópia válida do objeto solicitado foi encontrada no cache e também na memória, isto anulou o acesso ao disco.
- TCP_DENIED: o acesso foi negado para esta solicitação.
- TCP_OFFLINE_HIT: um objeto requisitado foi retornado ao cache durante o período que o solicitante estava *offline*. O modo *offline* nunca valida qualquer objeto.
- TCP_IMS_MISS: apagado, TCP_IMS_HIT usado no lugar deste.

- UDP_HIT: uma cópia válida do objeto solicitado foi encontrada no cache.
- UDP_MISS: o objeto solicitado não foi encontrado no cache.
- UDP_DENIED: o acesso foi negado para esta solicitação.
- UDP_INVALID: recepção de uma solicitação inválida.
- UDP_MISS_NOFETCH e UDP_RELOADING: a cache vizinha está carregando os seus dados a partir do disco (fase de inicializacão) e não permite que sejam feitos pedidos TCP de objetos *misses*.

  UDP_HIT_OBJ: objetos não estão disponíveis.

A.2 Códigos de Resposta do Protocolo HTTP

É com base no código retornado pelo protocolo HTTP, após ter transferido um objeto, que o *Squid* decide como armazenar o objeto [21]. As mensagens de resposta HTTP *iniciam com a Status Line*, que possui três campos: o número de versão do protocolo do servidor, o código de resposta e uma frase explicativa da resposta. Um pedido processado pode ter sucesso, falha ou ser redirecionado a outro servidor. Os erros podem ocorrer na requisição do cliente, conflitos, por excesso de tempo ou no servidor. Todos os tipos de respostas são agrupados em classes de respostas. Os códigos de resposta são divididos em 5 classes:

- *Informational 1xx*: são as respostas da classe informativa, estes códigos são utilizados para aplicações experimentais.
- *Successful 2xx*: códigos de respostas da classe de sucesso, a requisição do cliente foi recebida com sucesso pelo servidor e inclui a resposta apropriada com base no método pedido.
- *Redirection 3xx*: códigos de respostas da classe de redirecionamento. É usada para informar ao cliente que uma ação adicional é necessária para completar a requisição.
- *Client Error 4xx*: códigos de respostas da classe de erro do cliente. A requisição não pode ser atendida por erros cometidos pelo cliente, a requisição não deve ser repetida sem modificações.

- *Server Error 5xx*: códigos de respostas da classe de erro do servidor. O erro está situado no servidor, mudanças na requisição do cliente não corrigirão o problema. Estes códigos são utilizados quando o servidor souber que não pode realizar o pedido naquele momento.

Existem no total quarenta e um códigos de *status* de resposta no HTTP/1.1. O HTTP/1.1 manteve os dezesseis códigos de *status* de resposta do HTTP/1.0 [26] que ainda são significativos no HTTP/1.1. A seguir são mostrados os códigos de *status* de resposta do HTTP/1.1.

- 100 *Continue*: o cliente deve continuar com a requisição, permite a transmissão do pedido. Permite que o cliente saiba se o servidor será capaz de atender à expectativa do cliente com relação a sua requisição.
- 101 *Switching Protocols*: passa para outro protocolo. Serve como suporte para a atualização de outros protocolos.
- 200 *OK*: a requisição foi atendida com sucesso. A informação adicional retornada com a resposta depende do método de pedido.
- 201 *Created*: a requisição foi atendida e criado um novo recurso, normalmente utilizada para pedidos POST.
- 202 *Accepted*: a solicitação foi aceita mas ainda não foi processada totalmente. Mesmo podendo falhar mais tarde, a intenção desse código é permitir que o usuário continue com seu processo sem esperar o fim da operação no servidor de origem.
- 203 *Non-Authoritative Information*: os metadados retornados ainda não são definitivos, e foram obtidos de lugares fora do servidor origem. Provavelmente a informação obtida não é idêntica a que o servidor origem teria enviado.
- 204 *No Content*: o servidor informa que o processo de tratamento do pedido está finalizado e não é necessário mudança no que o usuário poderia ver.
- 205 *Reset Content*: o servidor informa que o processo de tratamento do pedido está finalizado e são necessárias complementações nas informações do cliente.

- 206 *Partial Content*: indica ao cliente que a resposta recebida não é uma resposta completa. É possível que o cliente solicite apenas os últimos *bytes* de um recurso.
- 300 *Multiple Choices*: é usado para indicar que um recurso pode ser selecionado de uma forma negociada a partir de uma série de representações possíveis, encontradas em diferentes locais.
- 301 *Moved Permanently*: é usado para indicar que o recurso solicitado possui um novo endereço.
- 302 *Found*: o recurso solicitado foi movido temporariamente para um endereço diferente, os clientes deverão continuar a usar o endereço antigo em futuras requisições.
- 303 *See Other*: o recurso solicitado pode ser encontrado em um endereço diferente.
- 304 *Not Modified*: é retornado se o horário de modificação de um recurso sendo validado não mudou desde o horário da última modificação, incluído no pedido.
- 305 *Use Proxy*: instrui a entidade solicitante a repetir o pedido por meio do proxy indicado.
- 306 *Unused*: utilizado em versões anteriores do HTTP 1.1.
- 307 *Temporary Redirect*: o recurso solicitado está temporariamente em um endereço diferente.
- 400 *Bad Request*: a sintaxe do pedido está incorreta ou não pode ser entendida pelo servidor.
- 401 *Unauthorized*: a requisição requer autenticação do cliente. Se o pedido não tiver a autorização necessária, então o servidor retorna o código de erro 401. Isso pode acontecer se o pedido não incluir qualquer informação de autorização ou se incluir informações de autorização inválidas.
- 402 *Payment Required*: reservado para uso futuro.
- 403 *Forbidden*: o pedido foi entendido pelo servidor, mas o servidor se recusou a atender.

- 404 *Not Found*: o servidor não conseguiu localizar o recurso solicitado.
- 405 *Method Not Allowed*: o método especificado na requisição não é permitido. É gerada uma resposta mais detalhada do problema para que o cliente corrija a solicitação.
- 406 *Not Acceptable*: retorna uma lista dos formatos disponíveis, evita que o cliente envie um pedido subseqüente para outro formato não disponível.
- 407 *Proxy Authentication Required*: indica que o recurso só está disponível a clientes autenticados.
- 408 *Request Timeout*: o cliente não produziu a requisição dentro do tempo que o servidor estava esperando. A requisição pode ser enviada sem modificações a qualquer momento.
- 409 *Conflict*: a requisição não pode ser completada por conflito no recurso solicitado. Se dois usuários tentam mudar um recurso por meio de PUT, o servidor de origem pode detectar isso como um problema e responder com uma resposta 409.
- 410 *Gone*: o recurso solicitado foi removido e seu novo local não pode ser determinado.
- 411 *Length Required*: permite que o cliente saiba que o servidor precisa saber o tamanho do conteúdo do corpo da mensagem antes de processar a requisição.
- 412 *Precondition Failed*: uma determinada pré-condição falha quando feito o teste no servidor, a atualização não é efetuada.
- 413 *Request Entity Too Large*: o servidor recusa a requisição por não estar habilitado a trabalhar com arquivo tão grande.
- 414 *Request-URI Too Long*: o servidor recusa a requisição por que o endereço solicitado é muito grande.
- 415 *Insupported Media Type*: é usado quando o formato do pedido do cliente está em um formato que não é aceito pelo servidor.
- 416 *Requested Range Not Satisfiable*: trabalha com requisições feitas sobre intervalos de *byte* quando nenhum dos intervalos puder ser retornardo através do recurso atual.

- 417 *Expectation Failed*: o servidor recebe uma requisição inesperada ou sabe que o servidor acima não poderá trabalhar com a requisição.
- 500 *Internal Server Error*: é retornada se o servidor não puder determinar a condição de erro exata.
- 501 *Not Implemented*: o servidor não suporta a funcionalidade requerida pela requisição.
- 502 *Bad Gateway*: é usado quando um servidor atualmente como um *proxy* ou *gateway* não consegue processar uma resposta de um outro servidor.
- 503 *Service Unavaliable*: o servidor não pode responder temporariamente, mas poderá responder mais tarde.
- 504 *Gateway Timeout*: o servidor, enquanto ativava um *gateway* ou *proxy*, não recebeu uma resposta em tempo hábil.
- 505 *HTTP Version Not Supported*: indica que o número de versão do protocolo na mensagem de pedido não é aceito pelo servidor.

A.3 Métodos de Pedido HTTP

Um método de pedido informa ao servidor HTTP qual ação deve ser executada sobre o recurso identificado pelo endereço do pedido. O método é aplicado ao recurso pelo servidor e a resposta é gerada. A resposta é composta por um código de resposta, dados sobre o recurso e os cabeçalhos de resposta complementares. O HTTP/1.1 define oito métodos que são descritos a seguir:

- GET: é aplicado ao recurso especificado no endereço que fez a requisição. A resposta gerada é o valor real do recurso, por exemplo, a transferência de um arquivo solicitado. Um pedido GET pode incluir argumentos construídos com base na entrada do usuário. A partir do HTTP/1.1 o método GET foi modificado para solicitar apenas partes de um recurso. Um pedido GET pode incluir um modificador de pedido, e isto pode resultar em uma ação diferente. Por exemplo, o método GET pode ser restrito a buscar um recurso apenas se a hora da última modificação do recurso solicitado for maior do que o valor especificado no cabeçalho *If-Modified-Since*.

- HEAD: pedido de apenas o cabeçalho HTTP para um endereço e não todo o documento. Os principais usos do método HEAD são depurar a implementação do servidor e determinar se um recurso foi alterado recentemente sem transferi-lo realmente.

- POST: é usado para transferir dados para o endereço especificado, serve para fazer as atualizações de recursos existentes ou para oferecer entrada para processos que manipulam dados. O usuário precisa ter autenticação necessária para modificar os recursos disponíveis.

- PUT: serve para armazenar dados em determinado endereço. O servidor origem processado um pedido PUT examina o endereço da solicitação para decidir se o pedido modificará um recurso existente ou criará um novo. O usuário precisa ter direitos de acesso para aplicar este método.

- CONNECT: reservado para uso futuro.

- DELETE: é usado para excluir os recursos do endereço identificado pela requisição do cliente. O método é fornecido para a exclusão de recursos remotamente. O usuário precisa ter direitos de acesso para aplicar este método.

- OPTIONS: serve para que o cliente verifique as informações a respeito das capacidades de um servidor.

- TRACE: permite que um cliente saiba o conteúdo da mensagem que foi realmente recebida pelo receptor.