Caracterização de Carga e Desempenho de um Servidor de Software

Cristina Duarte Murta, Aldo Monteiro do Nascimento

[1]Departamento de Informática – Universidade Federal do Paraná
Caixa Postal 19.081 – 81531-990 – Curitiba, PR – Brasil

Resumo. *Um servidor de software é um servidor Web que armazena código fonte e executável de programas que podem ser livremente copiados. Este artigo caracteriza a carga registrada em um servidor de software (mirror do SourceForge) no período de aproximadamente um dia, no qual foram servidos 3,5 Terabytes. O desempenho do servidor neste período é também analisado. Os resultados indicam que a carga de um servidor de software apresenta características bastante distintas da carga de um servidor de páginas Web. Vários aspectos relacionados ao desempenho do servidor são analisados e discutidos, em particular o equilíbrio entre a capacidade do servidor, a demanda e os recursos dos clientes.*

Abstract. *A software server is a Web server that stores source and executable code of programs that can be freely copied. In this paper we characterize the workload of a software server (SourceForge mirror) collected in a period of about one day, in which 3,5 Terabytes were served. The performance of the server in this period is analyzed. The results indicate that the software server workload is quite different from the workload of a Web page server. Several aspects related to the performance of the server are analyzed and discussed, in particular the dynamic balance between server capacity, server demand and client resources.*

1. Introdução

Caracterizar e entender a carga de sistemas computacionais é uma tarefa essencial em qualquer processo de análise ou avaliação de desempenho. O conhecimento adquirido nesta tarefa auxilia a identificação de problemas de desempenho, disponibilidade e confiabilidade do sistema. A caracterização de carga é também essencial para a construção de cargas sintéticas e para a proposição e validação de *benchmarks*. A evolução contínua dos sistemas, incluindo atualizações de hardware e de software, acréscimo de funcionalidades, e lançamento de novos produtos e aplicações, altera continuamente sua carga. Em conseqüência, a caracterização de carga deve ser também um processo contínuo.

Um servidor de software é um servidor que armazena código fonte e/ou código executável de programas que podem ser livremente copiados. A popularização do software livre contribuiu para a implantação de inúmeros servidores de software em todo o mundo. Um dos servidores de software mais conhecidos é o do *SourceForge* [SourceForge 2007]. O SourceForge é um sistema de gerenciamento de projetos de desenvolvimento de software, que oferece ferramentas para gerenciar o ciclo de desenvolvimento, incluindo controle de revisões e versões, ferramentas para comunicação entre os desenvolvedores e fóruns de discussão. Para tornar os projetos disponíveis aos

usuários, o SourceForge implementa o *File Release System*, que copia automaticamente projetos prontos em servidores *mirrors* espalhados pelo mundo. O número de arquivos copiados diariamente do SourceForge ultrapassa a casa dos milhões.

Um servidor de software pode ser implementado com um servidor Web, por exemplo, o Apache [Apache Web Server 2006]. No entanto, um servidor de software apresenta características bem distintas de um servidor de páginas Web no que diz respeito ao conjunto de arquivos armazenados. Enquanto páginas Web são compostas por vários arquivos pequenos, arquivos contendo código fonte ou executável são geralmente muito maiores. Além disso, para facilitar o processo de cópia do software desejado na área do cliente, dentre outras razões, todos os arquivos relacionados a um software ou pacote podem ser compactados em um único arquivo.

A caracterização da carga de servidores de páginas Web é bem conhecida [Arlitt and Williamson 1996, Arlitt and Jin 2000, Murta and Dutra 2004]. Observa-se grande variabilidade nos tamanhos das respostas às requisições, e a distribuição dos tamanhos apresenta características de cauda pesada [Crovella et al. 1998, Crovella and Krishnamurthy 2006]. A maior fração das requisições é para arquivos pequenos, mas observa-se a presença de poucos arquivos muito grandes, normalmente arquivos de áudio ou vídeo, ou mesmo código fonte ou executável. Além disso, servidores Web podem experimentar sobrecargas repentinas [Jung et al. 2002, Farah and Murta 2006].

Neste trabalho apresentamos uma caracterização das requisições feitas a um servidor de software, *mirror* do SourceForge [SourceForge UFPR Mirror 2006]. Este servidor serviu, no período de aproximadamente um dia, cerca de 3,5 Terabytes. Além da caracterização da carga, alguns aspectos do desempenho do servidor e de sua interação com os clientes e o sistema operacional são também analisados.

Não encontramos na literatura nenhuma caracterização de carga ou de desempenho de um servidor de software. No entanto, considerando o perceptível aumento do número de servidores de software em todo o mundo, em particular devido à expansão do software livre e do número de computadores, e as características peculiares dos arquivos armazenados neste tipo servidor, entendemos que o tema é relevante. Este estudo contribui para o entendimento de aspectos importantes da tarefa de servir arquivos muito grandes para uma grande diversidade de usuários via Internet, quando não há requisitos de tempo real.

Este artigo está organizado em cinco seções. A próxima seção discute os trabalhos relacionados. A Seção 3 apresenta a identificação e a caracterização da carga do servidor. A Seção 4 discute aspectos do desempenho do servidor, analisados a partir das informações coletadas no registro da carga. A Seção 5 conclui o artigo e apresenta direções para trabalhos futuros.

2. Trabalhos Relacionados

A caracterização de carga de servidores Web tem sido objeto de várias pesquisas, e é fundamental para entender o estado atual da aplicação mais visível da Internet, além de contribuir para o projeto de servidores e navegadores. O uso da Internet e da Web muda constantemente em função da evolução e diversificação dos sistemas, aplicações e software, o que altera também a carga, exigindo caracterização frequente ou contínua.

Alguns trabalhos são marcos nesta área. Um dos pioneiros foi o artigo que procurou identificar alguns invariantes na carga dos servidores Web [Arlitt and Williamson 1996]. Este trabalho apontou dez invariantes, analisando a carga de seis servidores Web, dentre eles o tamanho médio da transferência, a distribuição dos tamanhos, a concentração de referências, os tipos de arquivos, as características de clientes e domínios de origem das requisições. A distribuição de cauda pesada dos tamanhos dos arquivos em servidores Web é apontada como uma provável causa da auto-similaridade no tráfego da Internet [Crovella and Bestavros 1997, Crovella et al. 1998, Crovella and Krishnamurthy 2006], o que gera várias implicações para o projeto de roteadores e gerência de redes.

Um artigo recente [Faber et al. 2006] revisita o trabalho [Arlitt and Williamson 1996] analisando quatro cargas de servidores Web, três de servidores de dados científicos e uma de um servidor de ambiente acadêmico. Os autores analisam os invariantes identificados em 1996 e indicam que somente três são válidos para as cargas recentes analisadas. Os autores apontam ainda, como característica essencial dos servidores de dados científicos, a grande fração de conteúdo gerado dinamicamente, e de conteúdo acessado uma única vez.

Uma análise completa da carga de um servidor bastante requisitado, o do *site* da copa do mundo de futebol de 1998, foi descrita em [Arlitt and Jin 2000]. Este artigo caracteriza cerca de 1,3 bilhão de requisições feitas durante os 54 dias de realização do evento, totalizando aproximadamente 5 Terabytes transmitidos. Picos de carga foram observados durante a realização dos jogos e cerca de 88% das requisições foram para imagens. Os arquivos do *site* totalizaram 307 Mbytes, com tamanho médio de 15.524 bytes e mediana igual a 4.674 bytes. O maior arquivo tinha 61,2 Mbytes. Os resultados indicaram a necessidade de mecanismos mais eficientes de consistência de caches da Web.

Caracterizações da carga de sistemas de cache instalados em *backbones* da Internet, como os caches do NLANR [NLANR 2006], são úteis porque revelam características de agregados de páginas originadas de servidores e requisitadas por clientes de todo o mundo. Caracterizações de milhões de requisições coletadas em caches do NLANR indicam que os tamanhos dos arquivos estáticos de servidores Web são muito variáveis [Murta and Almeida 1998, Murta and Dutra 2004]. Os tamanhos médios dos arquivos, avaliados por cache, estão entre 13 e 20 Kbytes, e o coeficiente de variação dos tamanhos, calculado dividindo-se o desvio padrão pela média, está entre 9 e 39. Os tempos médios de serviço dos arquivos, também avaliados e comparados nestes trabalhos, estão entre 2 e 8 segundos, e os respectivos coeficientes de variação entre 10 e 27. Os tamanhos das respostas e os tempos de serviço podem ser modelados pela distribuição lognormal [Murta and Dutra 2004].

Similares aos servidores de software livre, os servidores de atualizações são mantidos por empresas de software proprietário tais como Microsoft, Adobe, Apple e Oracle, e inúmeras outras de pequeno e médio porte, para que seus clientes possam obter versões corrigidas e atualizadas (*patches*) de produtos adquiridos. A carga de um servidor de atualizações da Microsoft, correspondente a mais de um ano de coleta de dados, foi analisada recentemente [Gkantsidis et al. 2006]. Os resultados apresentam evidências dos comportamentos dos usuários e do tráfego gerado pelas requisições, que auxiliam o projeto de sistemas de disseminação de atualizações.

O projeto de servidores mais eficientes é um dos principais benefícios da caracterização, pois as características da carga revelam como os recursos do sistema são utilizados e como o projeto do sistema pode ser otimizado para tratar aquela carga específica [Crovella and Krishnamurthy 2006]. Por exemplo, em [Almeida et al. 2001], cargas de servidores Web com conteúdo multimídia são avaliadas com o objetivo de melhorar o projeto de servidores de mídia contínua, bem como identificar parâmetros para geração de carga sintética. Uma arquitetura que inclui monitoramento de desempenho para facilitar a avaliação do impacto da carga no servidor é descrita em [Maron et al. 2005]

3. Identificação e Caracterização da Carga

Para diminuir as ocorrências de sobrecarga, melhorar o tempo de obtenção (*download*) do pacote desejado e prover redundância, o SourceForge implementa servidores espelho (*mirrors*) em todo o mundo. Atualmente existem dezenas de servidores espelho espalhados pelo mundo. A carga analisada neste trabalho foi registrada no *mirror* do SourceForge [SourceForge 2007] localizado no Departamento de Informática da UFPR [SourceForge UFPR Mirror 2006], no período de 11 a 12 de outubro de 2006, totalizando cerca de 22 horas ininterruptas de registro, exatamente 79.920 segundos.

O ambiente computacional do servidor espelho inclui um processador AMD Athlon, com sistema operacional Linux, 3 Gbytes de memória principal e interface de rede Gigabit. O servidor está conectado por uma rede gigabit ao ponto de presença da RNP no Paraná (POP-PR) que, por sua vez, está conectado a instituições de ensino e pesquisa do Paraná e do país também por enlaces de alta velocidade. O servidor Web em execução é o Apache, versão 2.0. O conjunto de requisições HTTP recebido pelo servidor é denominado *carga do servidor* neste trabalho e é registrado pelo Apache em arquivos de *log*, seguindo o padrão *Common Log Format* [Apache Web Server 2006], acrescido de três diretivas.

Cada registro do *log* contém vários campos. Os campos utilizados neste trabalho estão descritos na Tabela 1. As informações registradas incluem a URL requisitada pelo cliente, o momento (dia, mês, ano, hora, minuto, segundo) em que a requisição foi finalizada, o tempo gasto em microssegundos para servir a requisição, o número de bytes servido, o código HTTP indicando sucesso ou erro na resposta e, finalmente, o número do processo e da *thread* vinculada ao processo que serviu a requisição. Outros campos foram registrados mas não são descritos pois não são avaliados neste trabalho. As diretivas acrescidas ao formato do *log* geraram os três últimos campos descritos na Tabela 1.

Inicialmente os registros foram analisados quanto ao método HTTP da requisição e ao código de sucesso da resposta. 99,81% das requisições foram feitas com o método GET, que indica a busca de um objeto designado pela URL. As demais requisições foram feitas com o método HEAD, que indica a consulta de informações sobre o objeto especificado sem sua transferência efetiva. Do total de requisições, 98,23% apresentou códigos de sucesso (200 e 206). Os registros restantes apresentaram códigos informando erro do cliente (4xx) e redirecionamento (3xx). Após filtrar os registros com erros, obtivemos 355.660 requisições, cuja análise é descrita a seguir.

Três aspectos da carga foram analisados: a taxa de requisições finalizadas pelo servidor, o tamanho dos arquivos servidos, ou seja, das respostas às requisições feitas pe-

Tabela 1. Informações do log utilizadas neste trabalho

Diretiva	Significado
%r	linha de requisição, inclui o método HTTP e a URL
%b	tamanho do objeto retornado
%t	tempo em que o servidor terminou a requisição
%s	HTTP *status code*
%D	tempo gasto para servir a requisição em microssegundos
%P	número do processo que serviu a requisição
%{tid}P	número da *thread* do processo P que serviu a requisição

los clientes, e a popularidade dos arquivos. Os resultados são apresentados nas subseções a seguir.

3.1. Taxa de requisições finalizadas pelo servidor

As informações contidas no *log* não são suficientes para inferir a taxa de chegada de requisições ao servidor, pois não há registro do momento de chegada de cada requisição. Apesar de conhecermos o tempo de serviço, não basta diminuir este valor do tempo de finalização, pois a requisição pode ter ficado em uma fila por algum tempo não conhecido, antes de ser efetivamente designada a um processo ou thread para execução. No entanto, podemos calcular a taxa de saída de requisições, ou seja, o *throughput* do servidor, dado que o momento em que a requisição foi finalizada é registrado. Além disso, com base no princípio do balanço de fluxo [Lazowska et al. 1984], podemos assumir que a taxa média de chegada de requisições é igual ao *throughput* médio do servidor, uma vez que requisições não são perdidas nem criadas no sistema.

Alguns índices estatísticos para a taxa de finalização de requisições são apresentados na Tabela 2. A unidade de medida é requisições por segundo (req/s). A mediana é 4 req/s e a média é 4,47. A variação do conjunto de valores em relação à média é dada pelo coeficiente de variação (COV), que é a razão entre o desvio padrão e a média. A taxa de saída é pouco variável (COV < 1). Observamos no período apenas um evento que pode ser considerado um pico de carga, com a finalização de 74 requisições em um segundo, o que corresponde a mais de dez vezes a taxa média.

Tabela 2. Índices estatísticos da taxa de finalização de requisições.

Índice	Menor	Mediana	Média	Maior	COV
Valor	0	4	4,47	74	0,59

3.2. Tamanho das Respostas

O segundo aspecto caracterizado é o tamanho das respostas às requisições feitas pelos clientes, isto é, o número de bytes servidos. A Tabela 3 apresenta alguns índices estatísticos para esta métrica. Observamos que o tamanho médio da resposta é cerca de 10 Mbytes e a mediana é 3 Mbytes. Estes valores são 3 ordens de grandeza maiores do que os mesmos índices de páginas Web, que apresentam mediana entre 2 e 5 Kbytes e média entre 13 e 21 Kbytes [Arlitt and Williamson 1996, Arlitt and Jin 2000, Crovella et al. 1998, Murta and Dutra 2004]. No entanto, diferentemente da caracterização dos tamanhos das

páginas Web, os tamanhos dos arquivos do servidor de software não apresentam grande variabilidade. Enquanto o coeficiente de variação dos tamanhos foi de apenas 3,36 para esta carga (Tabela 3), os mesmos trabalhos citados acima indicam COVs de no mínimo 9, chegando a 39, evidenciando maior variabilidade nos tamanhos dos arquivos que compõem páginas Web.

Tabela 3. Estatísticas dos tamanhos das respostas em bytes

Índice	Menor	Mediana	Média	Maior	COV
Valor	1	3.062.525	9.952.059	1.821.947.242	3,36

Os gráficos da freqüência acumulada (CDF) e da cauda da distribuição dos tamanhos (CCDF) são apresentados na Figura 1. O gráfico CCDF não evidencia presença de cauda pesada, uma vez que a probabilidade de observar arquivos grandes cai rapidamente para zero. Observamos que mais de 90% dos arquivos tem tamanho entre 10^4 e 10^8 bytes. Para comparação, no trabalho [Faber et al. 2006], a maioria dos arquivos solicitados tem tamanho entre 10^2 e 10^5 bytes, com evidência de cauda pesada. Os arquivos servidos no servidor de software são, portanto, muito maiores do que os servidos em servidores de páginas Web. Além disso, os tamanhos dos arquivos de software não são muito variáveis e não apresentam cauda pesada em sua distribuição.

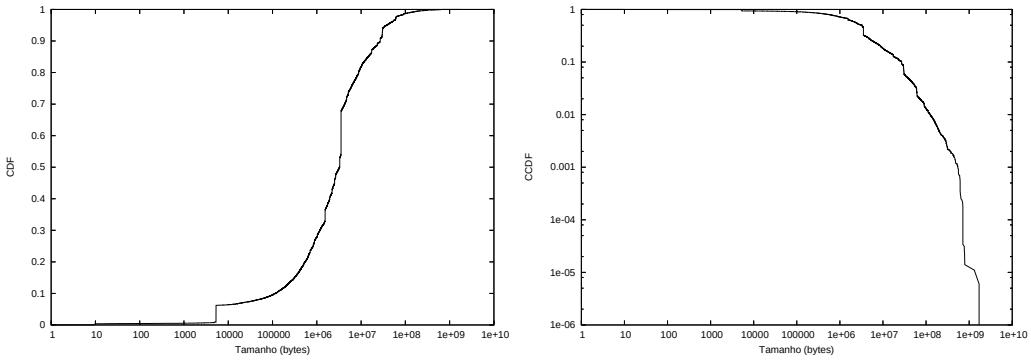

Figura 1. Distribuição acumulada e complemento da distribuição acumulada dos tamanhos dos arquivos servidos.

No período analisado neste artigo, de cerca de 22 horas, foram servidos 3,5 Terabytes, o que equivale a uma taxa média de serviço igual a 44,69 Mbytes/s ou 357 Mbits/s. O número médio de requisições por segundo foi 4,47. Para comparação, o *site* do servidor da copa do mundo de 1998 [Arlitt and Jin 2000] serviu quase 5 Terabytes no período de 54 dias, com média de 180 requisições por segundo, correspondentes a 680 Kbytes/s ou 5,44 Mbits/s. O servidor de software apresenta, portanto, características bem distintas do servidor de páginas.

3.3. Popularidade dos Arquivos

Um aspecto importante na caracterização de um conjunto de arquivos é a distribuição da popularidade do conjunto. O cálculo da popularidade deve ser feito com cuidado para evitar distorções. O protocolo HTTP 1.1 permite a transferência parcial do conteúdo requisitado se o cliente já possui cópia de parte do conteúdo. Uma cópia é feita parcialmente

se ocorre um erro durante o processo de transferência, por exemplo, se a conexão é interrompida. Assim, um objeto pode ser requisitado várias vezes e transferido em etapas até que a resposta seja completada.

Para evitar superestimativa da popularidade dos arquivos, consideramos nesta avaliação tanto o nome dos arquivos requisitados quanto o número de bytes transferidos. Identificamos arquivos com o mesmo nome e mesmo número de bytes, e contamos quantas vezes cada arquivo foi servido. Esta é uma abordagem conservadora pois não considera envio de conteúdo parcial. A Tabela 4 apresenta os dez arquivos mais populares no período analisado, em ordem de popularidade, incluindo o número de requisições feitas para cada arquivo, e o tamanho do arquivo em bytes.

Tabela 4. Os dez arquivos mais populares

Posição	No. de requisições	Tamanho (bytes)	Nome do Arquivo
1	47000	3.534.076	eMule0.47c-Installer.exe
2	18550	5.322	Azureus2.5.0.0.jar.torrent
3	12169	1.542.120	aresregular192_installer.exe
4	9517	3.308.767	Shareaza_2.2.1.0.exe
5	4166	2.519.716	eMulePlus-1.2.Installer.exe
6	3666	27.137.414	MarioF30.zip
7	3127	17.372.875	stellarium-0.8.2.exe
8	3087	2.206.322	eMule
9	2363	2.666.152	eMule0.47c.zip
10	2001	4.722.043	eMule0.47c-Sources.zip

A Figura 2 apresenta o gráfico da popularidade para todos os arquivos. O número de arquivos únicos no conjunto é de 144.104. O eixo x apresenta os arquivos em ordem de popularidade e o eixo y contém o número de requisições feitas a cada arquivo. Por exemplo, o arquivo mais popular foi requisitado 47.000 vezes, e o centésimo mais popular foi requisitado 153 vezes. Este gráfico segue a lei de Zipf [Adamic and Huberman 2002], com expoente bem próximo de 1.

A Figura 3 apresenta a concentração de referências. Para construir este gráfico, os arquivos são ordenados em ordem decrescente do número de referências, isto é, o número de vezes que cada um foi solicitado no período de observação. São calculadas então três métricas. A primeira é a fração do espaço de armazenamento que os arquivos mais referenciados necessitam. A segunda métrica é a fração de bytes que estes arquivos representam, no total de bytes servidos. A terceira métrica é a fração de requisições que são atendidas por estes arquivos. O gráfico da Figura 3 indica que os 10% mais populares respondem por 63% do total de arquivos servidos, 42% do total de bytes servidos e ocupam apenas 5,8% do espaço de armazenamento necessário para armazenar todos os arquivos.

Além disso, cabe ressaltar que os cem arquivos mais populares somam 741 Mbytes e os duzentos arquivos mais populares somam 1,426 Gbytes. Considerando que o servidor conta com 3 Gbytes de memória principal, observamos que centenas de arquivos populares podem ser armazenados na memória principal e servidos diretamente, sem necessitar de acesso a disco, o que contribui efetivamente para o bom desempenho do

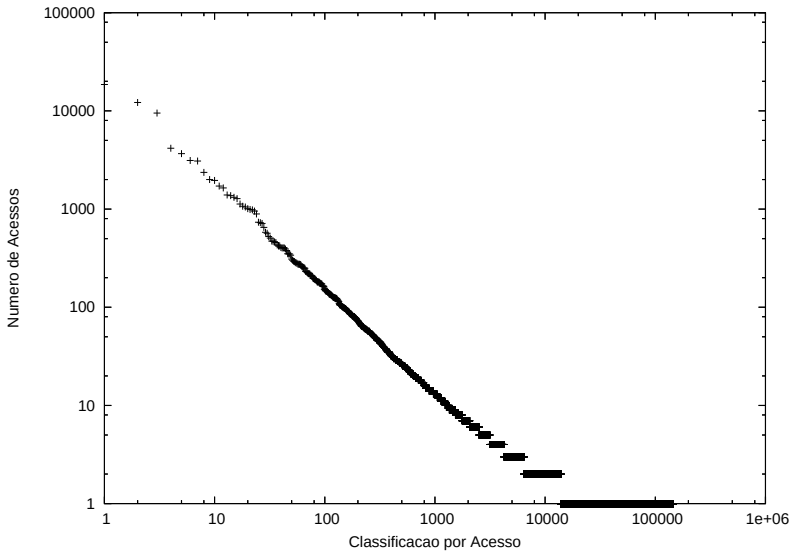

Figura 2. Classificacão dos arquivos em ordem de popularidade e o número de acessos feitos para cada arquivo.

servidor.

4. Caracterização do Desempenho

Os registros feitos no servidor contêm o tamanho das respostas em bytes, o tempo de serviço, o registro de tempo da finalização das requisições, e o número do processo e da *thread* que serviu a requisição. Estas informações permitem uma análise detalhada de alguns aspectos do desempenho do sistema.

4.1. Taxa de Serviço do Servidor

O *throughput* ou taxa de serviço do servidor refere-se à quantidade de bytes servida a cada segundo, calculada em Mbits/s. Este valor refere-se à vazão total produzida pelo servidor, resultado do atendimento concorrente de várias requisições. Os índices estatísticos desta métrica são apresentados na Tabela 5. A mediana é 163 Mbits/s e o *throughput* médio é 354 Mbits/s. Em 75% do tempo o servidor serviu 395 Mbits/s ou menos. A variabilidade da métrica, expressa pelo COV, é baixa.

Tabela 5. Throughput do servidor em Mbits/s

Índice	Menor	Quartil 1	Mediana	Quartil 3	Média	COV
Valor	0,0	59	163	395	354	1,85

Estes valores correspondem a menos da metade da capacidade nominal da interface de rede do sistema e da rede gigabit que conecta o servidor à Internet.

4.2. Análise dos Tempos de Serviço

O tempo de serviço de cada requisição, medido em microssegundos, é contado do momento de início do atendimento de uma requisição ao momento em que o último pacote é enviado para a interface de rede. Alguns índices estatísticos para o conjunto de tempos

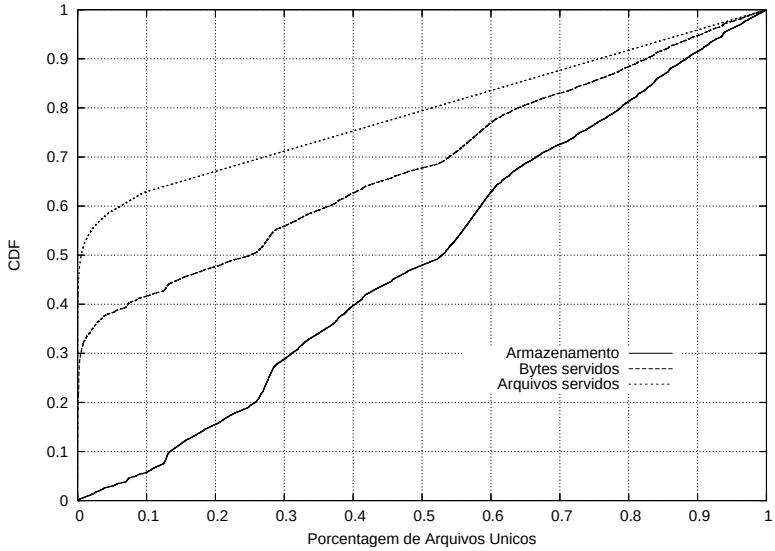

Figura 3. Concentração de referências: porcentagem de espaço, bytes servidos e requisições servidas pelos arquivos em ordem decrescente de popularidade

de serviço são apresentados na Tabela 6. Observamos que a mediana está em torno de 27 segundos, enquanto a média alcança 86 segundos. O maior valor, maior que 44.000 segundos, corresponde a cerca de 12 horas e trinta minutos. O coeficiente de variação desta métrica é um pouco superior ao coeficiente de variação dos tamanhos das requisições.

Tabela 6. Índices estatísticos dos tempos de resposta em microssegundos

Índice	Menor	Mediana	Média	Maior	COV
Valor	76	27.890.750	86.588.946	44.342.950.368	4,23

4.3. Análise das Taxas de Serviço por Requisição

A taxa de serviço foi calculada para cada requisição respondida com sucesso, dividindo-se o número de bytes servidos pelo tempo de serviço. Os índices estatísticos para esta métrica são apresentados na Tabela 7, em Kbits/s. Observamos que 25% das transferências são feitas a taxas inferiores a 259 Kbits/s e 75% são feitas a taxas inferiores a 16 Mbits/s. A mediana é 1,5 Mbits/s e a média é aproximadamente 59 Mbits/s. A variabilidade, indicada pelo COV, é bastante elevada. As razões da grande variabilidade desta métrica são investigadas nas próximas subseções.

Tabela 7. Índices estatísticos das taxas de serviço em Kbits/s

Índice	Menor	Quartil 1	Mediana	Quartil 3	Média	COV
Valor	0,007	259	1.507	16.114	58.992	19,92

4.3.1. Comparação entre Tempo de Serviço e Tamanho da Resposta

Um aspecto essencial do desempenho é a análise do tempo de serviço de uma tarefa no servidor em relação ao tamanho do serviço, no caso, a quantidade de bytes servida.

O serviço realizado pelo servidor consiste tão somente na transferência do arquivo solicitado. Nenhum processamento adicional é realizado no servidor, não há requisições dinâmicas nem scripts a serem executados.

A Figura 4 apresenta um gráfico no qual cada ponto representa o número de bytes servido (eixo x) em resposta a uma requisição e o tempo de serviço correspondente (eixo y). Os dois eixos estão em escala logarítmica. Observamos que os tempos de serviço para arquivos de mesmo tamanho são muito variáveis. Em especial, os tempos de serviço para arquivos maiores que 100 Kbytes abrangem uma faixa de valores que alcança 5 a 6 ordens de grandeza, de 10^4 μs, ou 0,01 s, a 10^{10} μs, ou seja, cerca de 10.000 segundos.

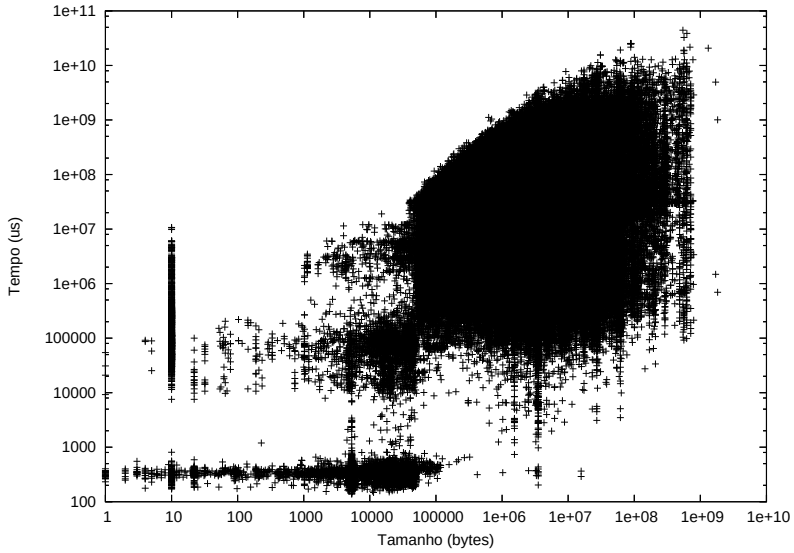

Figura 4. Tamanho e tempo de serviço para cada arquivo servido.

Esta variabilidade pode ter várias causas, entre as quais a sobrecarga no servidor, a variabilidade dos tamanhos das respostas e variabilidade nas condições de conectividade dos clientes. Conforme observado na Seção 3.2, os tamanhos das respostas não são muito variáveis. Os demais aspectos são investigados a seguir.

4.3.2. Análise dos Tempos de Serviço de um mesmo Arquivo

O *throughput* do sistema, bem abaixo de sua capacidade nominal, não indica que haja sobrecarga no servidor. Para confirmar este argumento, escolhemos alguns dos arquivos mais populares e plotamos os gráficos dos tempos de serviço para todas as requisições ao mesmo arquivo, ao longo do período de observação. A idéia é verificar a variabilidade dos tempos de serviço de diferentes requisições do mesmo arquivo, isto é, para servir o mesmo número de bytes. Se houver sobrecarga em alguns períodos, a variabilidade dos tempos de serviço será maior nestes períodos.

A Figura 5 apresenta os tempos de serviço para todas as requisições feitas ao arquivo Azureus2.5.0.0.jar.torrent (esq.), cujo tamanho é 5.322 bytes, e ao arquivo eMule0.47c-Installer.exe (dir.), cujo tamanho é 3.534.076 bytes. Em ambos os casos observamos que os tempos de serviço dos arquivos apresentam comportamento similar ao

longo do experimento, não demonstrando nenhuma correlação entre o tempo de serviço e o tempo do experimento, dado em segundos. Se o servidor experimentasse sobrecarga em algum período, os tempos seriam maiores ou, pelo menos, mais variáveis durante a sobrecarga, e isso não ocorre. A diminuição das requisições durante a madrugada pode ser observada no gráfico à direita da Figura 5, no intervalo entre 50.000 e 70.000 segundos.

Observamos também que os tempos de transmissão do arquivo de menor tamanho são muito menos variáveis, estão numa faixa bem mais estreita do que os tempos de transmissão do arquivo de maior tamanho. Como o arquivo é pequeno, o tempo gasto no servidor domina o tempo total de serviço, e o arquivo é servido é menos de 1 ms, exceto em poucos casos. A maioria absoluta dos valores está em apenas uma ordem de grandeza, cerca de centenas de microssegundos.

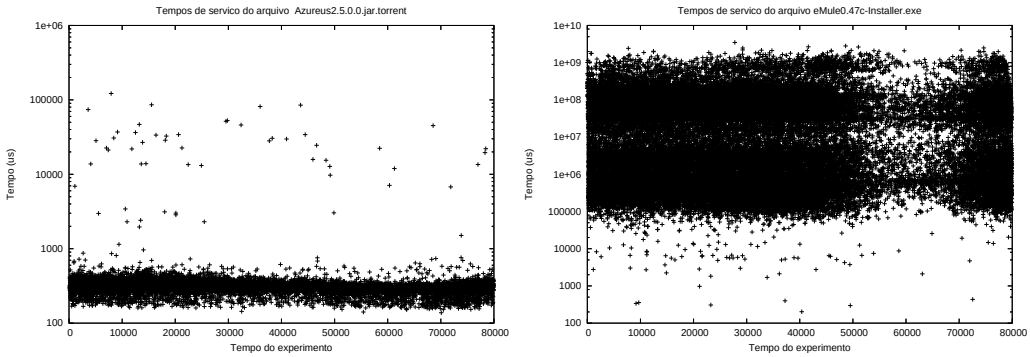

Figura 5. Tempos de serviço para todas as requisições feitas aos arquivos Azureus2.5.0.0.jar.torrent (esq.) e eMule0.47c-Installer.exe (dir.) durante o tempo de observação.

Por outro lado, os tempos de transmissão do arquivo de maior tamanho se espalham em cerca de 4 ou 5 ordens de grandeza. Para arquivos grandes, o tempo de transmissão do arquivo pela rede domina o tempo total de serviço, e o tempo de transmissão é limitado pela velocidade da conexão do cliente.

Para avaliar melhor as velocidades de transmissão dos arquivos grandes, calculamos os percentis da taxa de serviço por arquivo, para arquivos maiores do que 100 Kbytes. A Figura 6 apresenta estes dados. Observamos que cerca de 6% dos arquivos foram servidos a velocidades compatíveis com conexões realizadas via linha discada, 26% possivelmente utilizaram conexões de até 256 Kbits/s, 38% até 512 Kbits/s e 56% até 2 Mbits/s. Os demais foram servidos via enlaces mais rápidos, porém limitados a 1 Gbit/s, que é a velocidade do enlace do servidor à Internet.

Os números parecem superiores aos que se poderia esperar do acesso à Internet da população brasileira. No entanto, devemos observar que a UFPR, onde o servidor está hospedado, está conectada tanto internamente (entre os campi) quanto externamente, a várias universidades e institutos de pesquisa sediados na cidade de Curitiba, no estado do Paraná e no país, por meio de enlaces de alta velocidade [POP-PR 2007]. Computadores instalados em universidades e institutos de pesquisa estão possivelmente entre os principais clientes deste servidor.

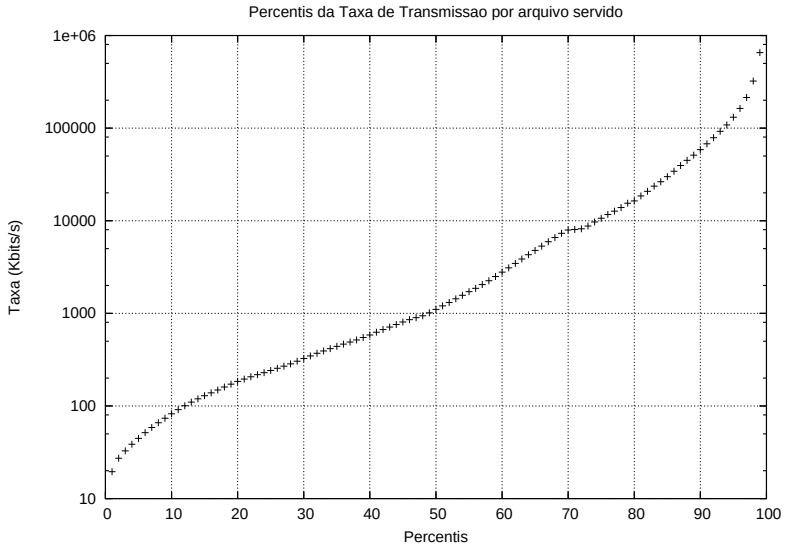

Figura 6. Percentis da taxa de serviço para arquivos maiores que 100 Kbytes.

4.4. Processos, Threads e Nível de Multiprogramação

O comportamento dos processos e das threads criadas pelo servidor Web para servir as requisições foi também analisado. O servidor Apache instalado implementa o módulo de multiprogramação MPM *worker* [Apache Web Server 2006], que cria um conjunto de processos, cada um com um número fixo de threads, para atender as requisições. Se necessário, novos processos são criados, com o mesmo número de threads, de acordo com a carga. A Tabela 8 indica os processos criados pelo servidor para atender as requisições, o número de threads utilizado em cada processo, o número de requisições atendidas por processo e o número médio de requisições atendidas por thread. Vale lembrar que o servidor já estava em execução antes do período desta coleta de dados, e assim continuou. O primeiro processo foi encerrado pelo servidor durante o período de experimento, e os três últimos foram criados no período final da coleta de dados. Isto explica o menor número de requisições atendidas por estes processos.

Tabela 8. Processos e threads utilizados para atender as requisições

Processo	Requisições	Threads	Req.por thread
10004	75237	256	294
10005	102555	256	401
10526	102786	256	402
11194	59719	256	233
22842	315	218	1,2
23512	10727	256	42
24095	4321	254	17

O gráfico da Figura 7 mostra o número de threads em atividade em um dos processos durante o período observado. Verificamos que, para cada processo, o número de threads atendendo requisições simultaneamente não alcança em nenhum momento o número de threads criado por processo (256), mesmo no caso deste processo, o mais ativo

no período. A estratégia do servidor é manter algumas threads ociosas e, portanto, disponíveis para atender novas requisições, evitando que uma requisição precise esperar pela criação de um novo processo para ser atendida.

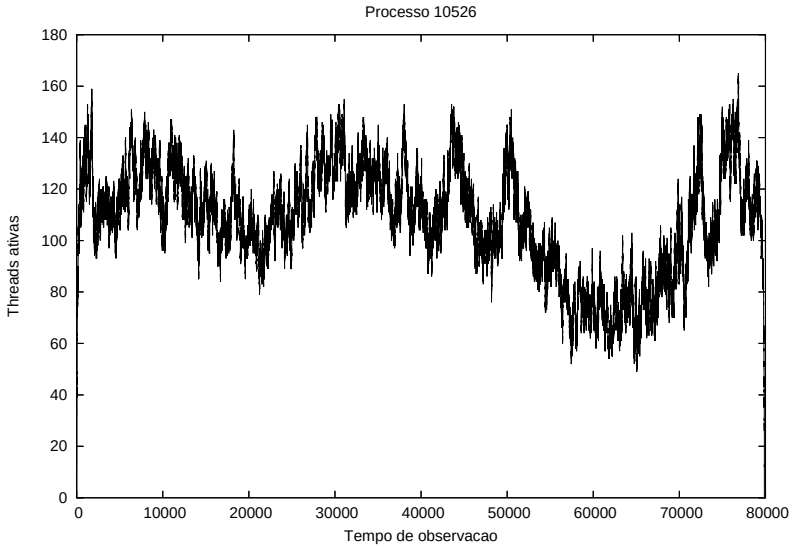

Figura 7. Threads ativas do processo 10526 durante o período de observação

O nível de multiprogramação do sistema pode ser calculado pela lei de Little [Lazowska et al. 1984], multiplicando-se a taxa média de serviço em req/s (4,47) pelo tempo médio de serviço (86,6 segundos). Assim, o número médio de requisições em atendimento no sistema é 387. O comportamento dos processos e das threads no servidor são também evidência de que o servidor não ficou sobrecarregado em nenhum momento.

4.5. Discussão

A análise apresentada revelou que a carga do servidor de software apresenta características bem distintas da carga de outros tipos de servidores, em especial da carga de servidores de páginas Web. Os arquivos são muito maiores, da ordem de Mbytes, em vez de Kbytes, e os tamanhos apresentam menor dispersão. As taxas de requisição não são muito variáveis, e o número médio de requisições por segundo é baixo. No entanto, a quantidade de bytes servidos é muito maior do que a observada nos demais servidores. O servidor demonstrou ter infra-estrutura adequada em termos de recursos (CPU, memória, disco e rede) para atender a carga, não apresentado evidência de sobrecarga. Servidores atuais são capazes de servir arquivos a taxas bastante elevadas, especialmente quando os arquivos são mantidos em memória principal. O servidor em questão é equipado com memória principal de 3 Gbytes, suficiente para armazenar fração significativa dos arquivos mais requisitados.

As taxas de serviço por arquivo são muito variáveis. As taxas de serviço dependem da velocidade de conexão dos clientes ao servidor, das condições do tráfego no período em que o arquivo é servido, além de vários fatores. Por exemplo, clientes de provedores de acesso são servidos via proxy do provedor, o que contribui para a latência da requisição. Diversas tecnologias de acesso são utilizadas para conectar o usuário à Internet, desde linhas de alta velocidade a acesso discado e redes em fio, incluindo serviços oferecidos por

provedores como conexões DSL e via cabo. Embora haja aumento contínuo na capacidade nominal de acesso à Internet, é razoável admitir que estas diferenças devem persistir pois estão relacionadas a aspectos econômicos e à distribuição da riqueza no mundo.

Estudos demonstram que cargas diferentes saturam recursos diferentes do servidor [Farah and Murta 2006]. Maior número de requisições pequenas requer mais recursos de software do servidor, como número de processos abertos, número de arquivos abertos, e número de sockets TCP. Requisições maiores normalmente saturam a rede.

Os resultados não indicam sobrecarga no servidor. Ainda assim, o servidor pode ser limitado pelas condições de conectividade dos clientes. Se o cliente é lento, o servidor precisa esperar, e não há meios para acelerar a resposta. Enquanto espera o cliente, o servidor mantém a conexão, com gasto de recursos como estruturas de dados, memória, sockets criados, threads. Esta situação demonstra o equilíbrio dinâmico entre a capacidade do servidor, a demanda e as condições de conectividade entre servidor e clientes. Nas condições analisadas, um aumento na carga pode mudar esta situação de estabilidade, gerando sobrecarga no servidor. Além disso, a melhoria das condições de conectividade dos clientes pode também alterar o quadro, sobrecarregando o servidor.

5. Conclusão

Resultados de caracterização de carga são informações essenciais para o projeto de sistemas. Este artigo analisa, possivelmente pela primeira vez, a carga de um servidor de software livre. Os resultados indicam que a carga é consideravelmente diferente das cargas de servidores Web analisados anteriormente, em diversos aspectos, exceto a popularidade dos arquivos servidos, que segue a lei de Zipf.

O projeto de servidores Web para este tipo de carga pode se beneficiar largamente deste estudo. Em particular, o escalonamento das requisições pode ser feito levando-se em consideração o tamanho da requisição e a capacidade do caminho entre cliente e servidor. Como as respostas são grandes, envolvem grande número de pacotes, e demoram vários segundos, o próprio servidor pode estimar a taxa de serviço de cada requisição, e escalonar a requisição de acordo. Esta situação é adequada para a implementação de políticas de escalonamento baseadas no tamanho da tarefa [Schroeder and Harchol-Balter 2006] e na velocidade da conexão [Murta and Corlassoli 2003].

Alguns aspectos registrados no *log* não foram analisados neste trabalho, por exemplo a origem das conexões, dada pelo endereço IP do cliente, e aspectos relacionados à localidade de referência das requisições. Estas análises são complementares a este trabalho e serão realizadas no futuro.

Agradecimentos

Os autores agradecem ao prof. Marcos Castilho pela cessão dos logs do *mirror* do SourceForge hospedado no Departamento de Informática da UFPR para esta análise.

Referências

Adamic, L. A. and Huberman, B. A. (2002). Zipf's law and the Internet. *Glottometrics*, 3:143–150.

Almeida, J. M., Krueger, J., Eager, D. L., and Vernon, M. K. (2001). Analysis of Educational Media Server Workloads. In *NOSSDAV '01: Proceedings of the 11th International Workshop on Network and Operating Systems Support for Digital Audio and Video*, pages 21–30, New York, NY, USA. ACM Press.

Apache Web Server (2006). http://www.apache.org.

Arlitt, M. and Jin, T. (2000). A Workload Characterization Study of the 1998 World Cup Web Site. *IEEE Network*, 14(3):30–37.

Arlitt, M. and Williamson, C. (1996). Web Server Workload Characterization: The Search for Invariants. In *Proc. 1996 ACM SIGMETRICS*, pages 126–137.

Crovella, M. and Krishnamurthy, B. (2006). *Internet Measurement: infrastructure, traffic and applications*. John Wiley & Sons, Ltd.

Crovella, M., Taqqu, M. S., and Bestavros, A. (1998). *A Practical Guide To Heavy Tails*, chapter Heavy-Tailed Probability Distributions in the World Wide Web, pages 3–26. Chapman & Hall, New York.

Crovella, M. E. and Bestavros, A. (1997). Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes. *IEEE/ACM Transactions on Networking*, 5(6):835–846.

Faber, A. M., Gupta, M., and Viecco, C. H. (2006). Revisiting Web Server Workload Invariants in the Context of Scientific Web Sites. In *SC '06: Proc. 2006 ACM/IEEE Conference on Supercomputing*, New York, NY, USA. ACM Press.

Farah, P. R. and Murta, C. D. (2006). TORÓ: Um Gerador de Sobrecarga para Servidores Web. In *Anais do XXVI Congresso da Sociedade Brasileira de Computação*, volume WPerformance.

Gkantsidis, C., Karagiannis, T., and Vojnovic, M. (2006). Planet Scale Software Updates. In *ACM SIGCOMM*, pages 423–434.

Jung, J., Krishnamurthy, B., and Rabinovich, M. (2002). Flash Crowds and Denial of Service Attacks: characterization and implications for CDNs and Web sites. In *WWW '02: Proceedings of the 11th International Conference on World Wide Web*, pages 293–304. ACM Press.

Lazowska, E. D., Zahorjan, J., Graham, G. S., and Sevcik, K. C. (1984). *Quantitative System Performance: Computer system analysis using queueing network models*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Maron, B., Chen, T., Vianney, D., Olszewski, B., Kunkel, S., and Mericas, A. (2005). Workload Characterization for the Design of Future Servers. In *Proc. IEEE International Workload Characterization Symposium*, pages 129 – 136.

Murta, C. D. and Almeida, V. A. F. (1998). Characterizing Response Time of WWW Caching Proxy Servers. In *WWC '98: Proc. IEEE Workshop on Workload Characterization, em conjunto com IEEE/ACM Micro*, Washington, DC, USA. IEEE Computer Society.

Murta, C. D. and Corlassoli, T. P. (2003). Fastest Connection First: A new scheduling policy for Web servers. In *Proceedings of the 18th International Teletraffic Congress (ITC-18)*.

Murta, C. D. and Dutra, G. N. (2004). Modeling HTTP Service Times. In *Proc. IEEE Global Telecommunications Conference, GLOBECOM '04*, volume 2, pages 972– 976.

NLANR (2006). National Laboratory for Applied Network Research. htpp://www.nlanr.net.

POP-PR (2007). Ponto de Presença da Rede Nacional de Pesquisa no Paraná. http://www.pop-pr.rnp.br.

Schroeder, B. and Harchol-Balter, M. (2006). Web servers under overload: How scheduling can help. *ACM Trans. Inter. Tech.*, 6(1):20–52.

SourceForge (2007). http://sourceforge.net.

SourceForge UFPR Mirror (2006). http://ufpr.dl.sourceforge.net.