Análise de Desempenho de um Servidor WWW

Cristina Duarte Murta [*] Jussara Marques de Almeida [†]
Virgílio A. F. Almeida [‡]

Resumo

Usuários da WWW desejam acesso rápido e fácil a todos os documentos disponíveis na rede. Portanto, o desempenho de servidores Web é uma questão cada vez mais importante. A latência de uma requisição Web, isto é, o tempo de resposta visto pelo usuário, é afetado pelo desempenho de três componentes: o cliente, o servidor e a rede que interliga clientes e servidores. Para melhorar a latência, uma instituição provedora pode atuar basicamente no servidor. Este artigo apresenta medidas de desempenho de um servidor Web e aponta os fatores que são obstáculos à obtenção de melhor desempenho.

Abstract

WWW users want fast and easy access to all documents available on the net. Thus, Web server performance is becoming increasingly important. The perceived Web latency i.e., the request response time, is affected by the performance of three components: the client, the server and the network that connects clients and servers. To improve the Web latency, the provider institution can act basically in the server. In this paper, we present measures of a Web server performance and point out the main bottlenecks.

1 Introdução

O crescimento fantástico da *World Wide Web* (WWW ou Web) [2] em termos de acesso e de criação de novos *sites* tem sido alardeado não apenas pelas publicações técnicas [10] mas também pela mídia em geral. Este crescimento tem incentivado pesquisas sobre tópicos relacionados à Web e sua infraestrutura de funcionamento. Estas pesquisas estão, na maioria dos casos, voltadas ou para aspectos do cliente ou para aspectos do servidor, refletindo a arquitetura cliente–servidor do sistema.

[*]Professora Assistente do Departamento de Informática da Universidade Federal do Paraná. Em doutoramento no DCC/UFMG, com bolsa da CAPES. E-mail: `cristina@dcc.ufmg.br`

[†]Aluna de Mestrado, Departamento de Ciência da Computação, Universidade Federal de Minas Gerais, caixa postal 702, cep 31270-010. Belo Horizonte, Minas Gerais. E-mail: `jussara@dcc.ufmg.br`

[‡]Professor Titular do DCC/UFMG e atualmente Professor Visitante no Computer Science Department, Boston University, 111 Cummington St, Boston, MA 02215. E-mail: `virgilio@cs.bu.edu`

Dentre os tópicos mais investigados podemos citar: monitoração e análise de desempenho [11], caracterização da carga [1, 7], melhoramentos nos protocolos necessários à interação [14] e estratégias para redução da latência das requisições como *cache* de clientes [3] e de servidores [6].

Usuários Web desejam acesso rápido e fácil aos documentos disponíveis na rede. O tempo de resposta a uma requisição do usuário depende de vários componentes da WWW. Um componente essencial no tempo de resposta de uma requisição é o servidor Web. Parte considerável do tempo de resposta é gasto neste servidor e muitas requisições são recusadas a partir dele. Assim, o desempenho de um servidor é fundamental para o funcionamento satisfatório da WWW.

Este trabalho analisa detalhadamente o desempenho de um servidor Web no ambiente Pentium/Windows NT. O objetivo é investigar como se comporta um servidor Web, onde e porque os principais recursos são gastos e quais recursos do ambiente limitam seu desempenho em função do volume e do tipo da carga imposta. Os experimentos consistiram em gerar requisições direcionadas ao servidor instalado. Como resultados, foram coletados dados que expressam medidas de desempenho do lado do cliente e do servidor. Os dados da máquina NT coletados durante a operação do servidor foram obtidos através de uma ferramenta fornecida pelo próprio Windows NT, denominada *Performance Monitor* [4]. As requisições foram geradas por um *benchmark* denominado Webstone [16], cujos resultados refletiram o desempenho observado do lado do cliente. Estas medidas de desempenho do cliente e do servidor foram analisadas e comparadas. Os resultados mostram que, para uma carga típica, o servidor analisado tem seu desempenho limitado pela utilização do processador e não pela utilização de disco ou memória.

Os trabalhos sobre análise de desempenho de servidores Web publicados até agora apresentam comparações de desempenho entre dois ou mais servidores instalados em vários ambientes [5, 9, 11]. Esta característica comparativa visa determinar qual servidor tem o melhor desempenho em termos da taxa de atendimento e/ou tempo de resposta. Nosso enfoque não é comparativo mas analítico. Uma análise detalhada do comportamento exibido pelo servidor em função das requisições dos clientes é feita. A comparação apresentada é feita entre os resultados de desempenho obtidos por ambas as partes durante a **mesma sessão** de interação.

Este artigo contém cinco seções, das quais esta é a primeira. A segunda seção apresenta alguns conceitos básicos no ambiente Web e as métricas de desempenho mais utilizadas neste ambiente. A seção 3 descreve o ambiente de experimentação e os testes realizados. A seção seguinte apresenta e analisa os resultados obtidos. A seção 5 traz as conclusões.

2 O Ambiente Web

2.1 Conceitos Básicos

O projeto da Web tem como objetivo criar ferramentas e sistemas que permitam o acesso e a recuperação de informação num grande hipertexto global [2]. Estas informações, espalhadas por várias partes do mundo, podem estar em diferentes formatos: áudio, vídeo, texto e gráficos. A Web é baseada no modelo cliente–servidor. Um **servidor** Web é um programa que aceita conexões com o objetivo de servir e responder requisições de acesso a informações. Um **cliente** Web é um

programa que estabelece conexões com o propósito de enviar requisições feitas por usuários a um servidor.

Um arquivo ou serviço é representado na Web através de uma URL – *Uniform Resource Locator* [15]. Uma URL especifica um método de acesso e o endereço do computador onde o arquivo ou serviço solicitado se encontra, entre outras informações. Os métodos de acesso podem ser **http**, **ftp**, **telnet**, **gopher**, **wais**, **mailto** e **news**. Este trabalho trata especificamente de requisições **http**.

A navegação na Web é feita através de documentos hipermídia, ou páginas, que podem conter *links* que representam conexões com outros arquivos ou serviços. Páginas são construídas usando uma linguagem chamada HTML – *HyperText Markup Language*, que define os arquivos que devem aparecer na página, seus formatos, sua localização e *links* para outras páginas.

A comunicação entre clientes e servidores fica a cargo do protocolo HTTP – *Hypertext Transfer Protocol* [15]. Esta comunicação é sempre na forma de pares requisição–resposta e sempre iniciada pelo cliente. O HTTP é um protocolo genérico, orientado a objetos, executado na camada de aplicação, que usa TCP como protocolo da camada de transporte. Uma interação HTTP é mostrada na figura 1. Inicialmente, o cliente envia uma mensagem de sincronização para o servidor pedindo abertura de conexão. O servidor responde aceitando ou não este pedido. Se o servidor aceitar, a conexão é estabelecida. Uma recusa pode indicar que o servidor está sobrecarregado e não consegue tratar os pedidos de conexão que chegam. Normalmente, existe um número máximo de pedidos pendentes. Um pedido de conexão pode ser recusado se este limite tiver sido atingido.

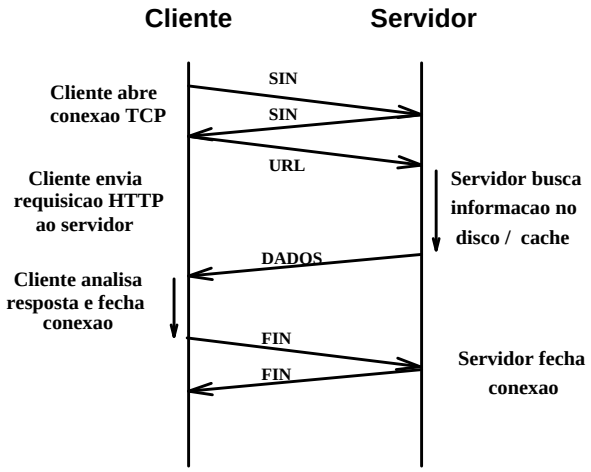

Figura 1: Troca de mensagens entre cliente e servidor durante uma interação HTTP

Uma vez estabelecida a conexão, o cliente envia uma mensagem com a requisição feita pelo usuário, representada por uma URL. Esta mensagem HTTP é enviada através da rede para o servidor. Antes de atingir o meio físico de transmissão, um ou mais pacotes TCP são montados e submetidos à interface de rede. Cada um destes pacotes chega à interface de rede da máquina servidora causando uma interrupção. Esta interrupção é tratada pelo sistema operacional que extrai a mensagem HTTP e a envia para o porto dedicado ao servidor WWW. Este realiza um processamento sobre a mensagem recebida para saber qual a informação requisitada. O servidor busca esta informação no disco local.

Algumas vezes, parte da informação a ser retornada ao usuário exige do servidor algum processamento local. Isto pode ser devido à execução de aplicações CGI – *Common Gateway Interface*, que são programas executados em tempo real e que geram informação dinamicamente. Uma mensagem HTTP de resposta é, então, enviada ao cliente que, ao recebê-la, retorna um pedido de fechamento de conexão. Do ponto de vista do cliente, a conexão já está fechada. O servidor envia, finalmente, uma resposta de confirmação do fechamento de conexão.

Quando o cliente recebe a informação pedida, é necessário verificar se os dados são válidos, através de informações contidas em campos do cabeçalho da mensagem. Somente após esta verificação, o cliente pode mostrar o documento para o usuário. Para tanto, ele precisa saber qual o tipo e o formato do arquivo. Esta informação também é obtida no cabeçalho da mensagem HTTP. Este processamento impõe um atraso no cliente que é refletido no tempo de resposta observado pelo usuário.

2.2 Métricas de Desempenho

O desempenho de um servidor pode ser avaliado pelas seguintes métricas: **throughput** ou taxa de processamento (X_s) e **tempo de resposta** (R_s). O primeiro representa a taxa total de dados que o servidor pode produzir, medido em KBytes/s, ou em número de conexões atendidas por segundo, HTTPreqs[1]/s. O tempo de resposta do servidor para uma requisição (R_s) corresponde ao intervalo de tempo desde o instante em que a requisição chegou à interface de rede da máquina até o momento em que o servidor liberou a informação para a rede.

É preciso fazer uma distinção entre tempo de resposta do servidor e tempo de resposta do usuário. O tempo de resposta percebido pelo usuário, ou **latência** de uma requisição, é afetado pelo desempenho de três componentes: o cliente, o servidor e a rede que interliga cliente e servidor. O cliente impõe um atraso devido ao tempo gasto pelo *browser* para mostrar o documento pedido pelo usuário, considerando os diferentes tipos de mídias possíveis (T_c). A latência da rede representa o tempo necessário para prover o acesso remoto mais o tempo para a transmissão dos dados pedidos (T_r). O atraso no servidor é causado pelo processamento da requisição e liberação da informação pedida (R_s). A latência de uma requisição HTTP (L_{http}) pode então ser dada pela seguinte expressão: $L_{http} = T_c + T_r + R_s$ Para melhorar a latência de uma requisição, uma instituição interessada em tornar disponíveis informações na Web pode contribuir principalmente na parte deste tempo que se refere ao servidor, uma vez que os atrasos na rede e no cliente podem fugir à sua atuação.

O desempenho de um servidor Web depende de quatro fatores: a plataforma de hardware, o ambiente de software (sistema operacional, subsistema de gerência de arquivos, gerenciador de banco de dados, etc), o *daemon* HTTP e a carga. Como o protocolo HTTP utiliza o protocolo TCP na camada inferior, a implementação TCP/IP do sistema operacional da máquina servidora tem papel importante no desempenho do sistema. Além das diferenças de desempenho entre implementações distintas, uma mesma implementação pode ter parâmetros diversos para variáveis, tais como tamanho da fila de requisições pendentes, que influenciam diretamente a latência do sistema.

[1] requisições HTTP

A carga de trabalho de um servidor Web corresponde ao conjunto de requisições HTTP recebidas pelo servidor durante o período de observação do sistema. Parâmetros como tamanho e tipo da informação requisitada são muito importantes. Para alguns servidores Web, uma parte significativa das requisições HTTP resulta em execução em tempo real de programas CGI. Neste caso, o desempenho de um servidor depende do desempenho destes programas. O tamanho dos arquivos transmitidos do servidor para o cliente também influi no tempo de resposta. Arquivos maiores terão uma latência na rede maior.

Para melhorar o desempenho de um servidor Web é fundamental um entendimento sólido das características de uma carga típica. Vários trabalhos apresentam propostas de caracterização desta carga. Os principais aspectos pesquisados são tipo de arquivo, tamanho médio da transferência e concentração das referências. O trabalho [1] apresenta dez invariantes de cargas de servidores Web. Estes invariantes caracterizam transações típicas neste ambiente. Por exemplo, 90% das requisições são para arquivo HTML ou imagens; o tamanho médio de arquivo transferido é em torno de 12 KBytes e 10% dos arquivos disponíveis correspondem a 90% das requisições de um servidor e a 90% dos bytes transferidos. Portanto, um acesso típico pode se beneficiar do uso de *cache* no servidor, devido ao tamanho pequeno dos arquivos acessados e à boa localidade de referência.

3 Descrição dos Experimentos

Os experimentos foram realizados numa máquina dedicada ao servidor com as seguintes características: processador Pentium 100 MHz, 16 MBytes de memória, 256 KBytes de cache, sistema operacional Windows NT Workstation 3.5 com TCP/IP e conexão Ethernet. O software para servidor Web foi o EMWAC HTTPs produzido pelo *European Microsoft Windows NT Academic Centre* [8]. Este servidor implementa o protocolo HTTP/1.0 e foi projetado para para executar como um processo servidor denominado *https* sobre o sistema operacional Windows NT. A plataforma servidora foi ligada via uma rede Ethernet não dedicada a uma máquina Unix, a partir da qual foram geradas requisições HTTP por um *benchmark* sintético chamado WebStone [16]. Esta máquina é uma SparcStation 20 com 256 MBytes de RAM e sistema operacional SunOS 5.4.

Além das informações de desempenho obtidas pelo WebStone, dados sobre utilização dos recursos processador, memória, disco e rede pelo servidor são bastante importantes para a análise de seu comportamento. Tais informações foram obtidas a partir de uma ferramenta disponível no Windows NT chamada *Performance Monitor* [4]. Esta ferramenta faz monitoração de desempenho, colhendo informação sobre o comportamento de 22 objetos entre eles processos, *threads*, memória, processador, discos e outros recursos da máquina. Esta informação é exibida de maneiras diversas como gráficos e relatórios. Para cada objeto, o *Performance Monitor* oferece um conjunto de contadores. Um contador é uma entidade para a qual dados de desempenho estão disponíveis. Existem centenas de contadores. Os objetos monitorados durante os experimentos foram memória, disco físico, processador e processos. A descrição dos contadores utilizados em cada objeto é mostrada na tabela 1.

WebStone gera um conjunto de requisições de páginas que são submetidas a um servidor Web e mede o desempenho do conjunto programa servidor e plataforma de

Objeto	**Contador**	**Descrição**
Memória	*Pages/s*	Número de páginas transferidas de ou para o disco.
Disco Físico	% *Disk Time*	Porcentagem do tempo total que o disco foi utilizado.
Processador	% *Processor Time*	Porcentagem do tempo total que o processador foi utilizado.
	Interrupts/s	Taxa de interrupções de dispositivos.
Processos	% *Processor Time* do *https*	Porcentagem do tempo total que o *https* usou o processador.
	Thread Count do *https*	Número de *threads* ativas do *https*.

Tabela 1: Descrição dos objetos e contadores monitorados

software e hardware. Webstone opera da seguinte maneira: existe um processo mestre denominado *Webmaster* que cria local ou remotamente um número pré-definido de processos clientes que geram, de forma independente entre si, tráfego HTTP para o servidor Web em teste. Este tráfego é também pré-definido em um arquivo de configuração. Após o término da execução de todos os clientes, o *Webmaster* reúne os dados coletados e gera um relatório do desempenho do experimento. Webstone é projetado para executar por um período específico de tempo. O número de iterações também pode ser determinado. O número de processos clientes por máquina é limitado somente pela sua memória. Para os experimentos realizados, o Webmaster e os processos clientes foram disparados na mesma máquina Unix. O período de teste foi de 5 minutos, e os resultados apresentados são valores médios de 5 experimentos.

Os principais resultados que o Webstone fornece são o *throughput* medido em Kbytes por segundo e a latência. Outras medidas importantes são a taxa de conexão e *Little's Load Factor*, derivado da Lei de Little [12]. Este fator reflete o grau de paralelismo no processamento do servidor, isto é, o número médio de requisições que o servidor atende em um dado instante. O *Little's Load Factor* é calculado por $LLF = \frac{tempo\ acumulado}{tempo\ de\ teste}$, onde *tempo acumulado* é a soma das latências medidas de todas as conexões abertas e fechadas com sucesso por todos os clientes e *tempo de teste* é a duração da execução do *benchmark*. Idealmente, o *Little's Load Factor* deve ser igual ao número de processos clientes. Um valor inferior indica que o servidor está sobrecarregado e que alguns clientes não estão sendo servidos antes que o prazo de atendimento seja expirado. O *Little's Load Factor* também pode ser visto segundo a Lei de Little: $LLF = \lambda * \overline{L}_{http}$, onde λ é a taxa de conexões aceitas pelo servidor. Logo, ele reflete quanto tempo foi gasto pelo servidor apenas para executar as requisições concluídas sem incluir erros e *overhead*. A latência média fornecida pelo WebStone, $\overline{L}_{http}$, é calculada pela expressão: $\overline{L}_{http} = \frac{tempo\ acumulado}{número\ total\ de\ conexões\ com\ sucesso}$ Observando as expressões para L_{http} e para $\overline{L}_{http}$, pode-se compreender a importância do tempo de resposta do servidor na composição da latência.

A carga de trabalho sintética imposta ao servidor é definida pelo número de clientes e pelas características do arquivo de configuração. Este arquivo é composto por um conjunto de páginas cujo acesso é definido por um conjunto de probabilidades. Cada página, por sua vez, contém um conjunto de arquivos de diferentes tamanhos. Uma requisição a uma página representa uma requisição a cada um de

seus arquivos.

Os experimentos foram feitos usando duas cargas diferentes (A e B), cujas principais características são mostradas na tabela 2. Para carga A, 94% dos arquivos acessados são menores que 50 KBytes. Para a carga B, os tamanhos dos arquivos estão entre 1 e 200 KBytes com igual probabilidade de acesso.

Característica	**Carga A**	**Carga B**
Número de arquivos	18	180
Tamanho total dos arquivos	298 KBytes	5 MBytes
Tamanho médio do arquivo	15 KBytes	27 KBytes

Tabela 2: Características das cargas utilizadas

4 Resultados

Os resultados apresentados neste trabalho referem-se às medições obtidas através de monitoração dos componentes extremos da arquitetura da Web: cliente e servidor. Na visão do servidor, a ferramenta *Performance Monitor* registrou o uso dos recursos e o comportamento de *threads* e processos. Do ponto de vista do cliente, o Webstone mediu o tempo de latência de requisições HTTP, bem como a eficiência do sistema (LLF).

4.1 Taxa de Processamento e Utilização

A figura 2 mostra os resultados da taxa de processamento X_s, em (KBytes/s), e taxa de conexão, em HTTPops/s, obtidos do lado do cliente, pelo Webstone, e os resultados de utilização do processador medidos no servidor para as cargas A e B. A utilização do processador foi observada em dois níveis, utilização total e utilização pelo processo *https*. Os gráficos são apresentados em função do número de clientes. O gráfico da esquerda mostra que até 20 clientes, a taxa de conexão cresce à medida que aumentamos o número de clientes. A seguir, a taxa de conexão decresce devido à carga no processador, cuja utilização atinge quase 100%. A taxa de conexão alcança seu valor máximo quando o processador não pode mais tratar requisições, isto é, fica saturado.

As curvas de *throughput* e taxa de conexão têm comportamentos muito parecidos. À medida que o número de clientes aumenta, o número de pacotes TCP/IP que chegam ao servidor também aumenta. Pode-se notar que a curva de utilização do processador pelo *https* tem derivada menor que a curva de utilização total do processador no intervalo entre 10 clientes e o ponto de saturação. Uma vez que o processador tem que tratar todas as interrupções geradas pela chegada de pacote à interface de rede, o uso de processador pelo processo *https* cresce mais suavemente que a utilização total do processador, até atingir o ponto em que o processador está sobrecarregado e não consegue tratar as requisições. A partir deste ponto, o uso do processador pelo *https* diminui embora o uso total do processador se mantenha

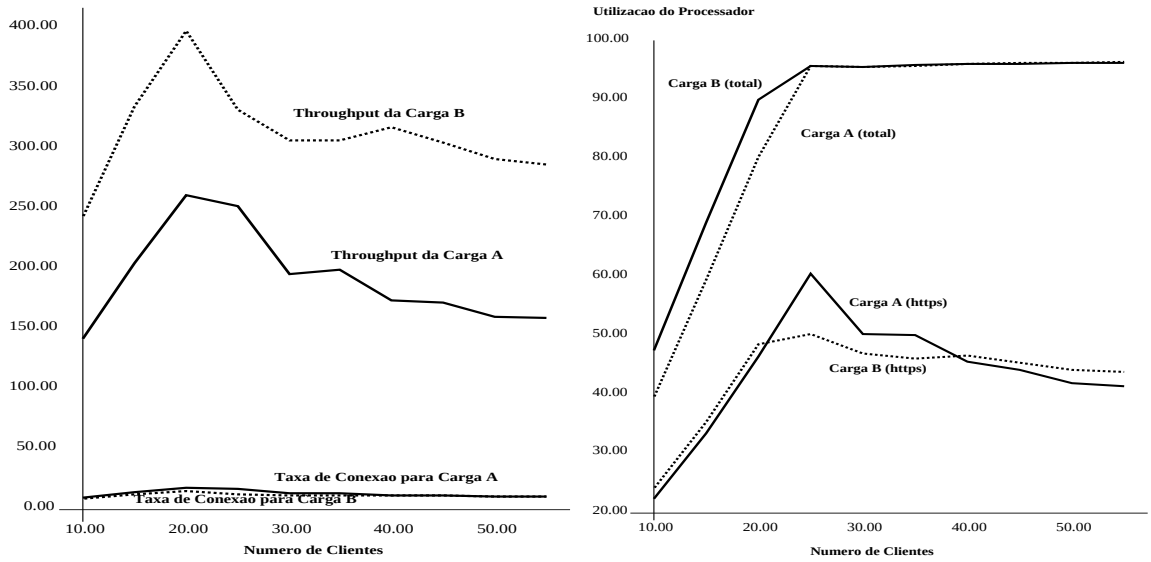

Figura 2: Medidas do cliente e do servidor: taxa de conexão (HTTPops/s), *throughput* (KBytes/s) e utilização do processador total e para o *https* para as duas cargas

constante. A taxa de conexão também diminui e, como consequência, o *throughput*. A diferença entre utilização total do processador e utilização do mesmo pelo *https* pode ser entendida como o processamento feito pelo sistema operacional para execução de chamadas ao sistema, em especial, para tratamento de interrupções. Apesar das curvas de taxa de conexão para cargas A e B serem bem próximas, o *throughput* para carga B é bem maior, uma vez que o tamanho médio de arquivo para esta carga é maior.

Os pontos de saturação indicados nos gráficos não são iguais: o gráfico da esquerda (visão do cliente) indica saturação com 20 clientes, ponto de máximo da taxa de conexão e *throughput*. Já o gráfico da direita (visão do servidor) indica saturação com 25 clientes, ponto de máxima utilização do processador pelo *https*. Isto pode ser explicado pelo fato de que nem tudo o que o servidor consegue processar, o cliente consegue receber. Existe um meio físico sujeito a erros entre servidor e cliente. Um arquivo HTML perfeitamente recuperado pelo servidor pode chegar corrompido ao cliente devido a algum erro na rede ou nas interfaces de rede do servidor e do cliente. Este erro é descoberto pelo cliente, quando este realiza o processamento final no pacote HTTP recebido, verificando a validade do arquivo nas informações do cabeçalho. Esta conexão não é contabilizada pelo WebStone; mas a utilização do processador no servidor é refletida nos contadores.

4.2 Latência e Taxa de Interrupção

Outra questão importante é a latência. Na figura 3, são apresentadas as curvas de latência, dada pelo Webstone, e da taxa de interrupções, monitorada no servidor, ambas em função do número de clientes. A curva de tempo de resposta, como esperado, aumenta com o número de clientes. Pode-se observar que as duas curvas do gráfico da direita têm o mesmo comportamento. Acima de 35 clientes o comportamento das curvas fica irregular. Acreditamos que isso se deve à sobrecarga do

processador ao atender as interrupções. Devido a essa sobrecarga, muitos pacotes TCP/IP são perdidos, e as correspondentes interrupções não são geradas. Observe que as duas curvas se cruzam por volta de 40 clientes. O mesmo ocorre com as curvas de tempo de resposta de utilização do processador para *https*. Depois do ponto de cruzamento, a taxa de interrupção para carga A se torna maior que a para a carga B, e a curva correspondente para a utilização de processador pelo *https* se tornou inferior para a carga A. Isto se deve ao fato de que o processador tem que tratar mais interrupções para a carga A; logo, a porcentagem de utilização do processador para esta carga é inferior à porcentagem para a carga B, fixado o número de clientes. Como uma outra consequência, o tempo de resposta para a carga A é maior que para a carga B.

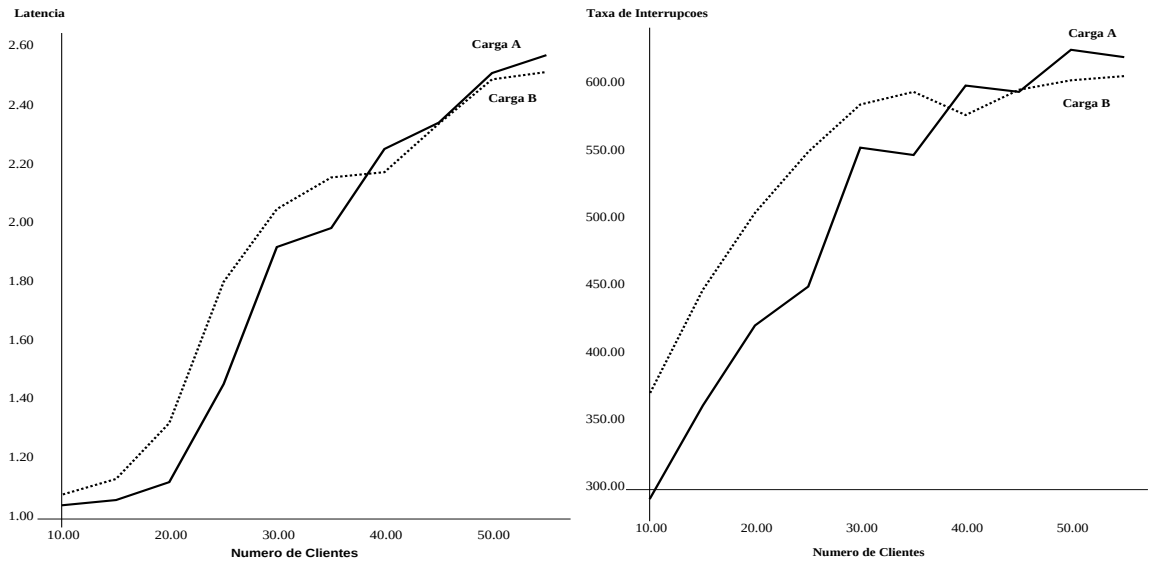

Figura 3: Medidas do cliente e do servidor: latência (segundos) e taxa de interrupções para as duas cargas

4.3 Disco e Memória

A figura 4 apresenta o comportamento do disco e da memória. As curvas de utilização do disco e de *bandwidth* são similares. O tempo de utilização do disco fica em torno de 0.9% para a carga A e 9% para a B. A taxa de páginas trocadas entre disco e memória também foi muito baixa.

Estes resultados indicam que disco e memória não provocam contenção na operação do servidor Web. Isto foi constatado no ambiente EMWAC HTTPs/Windows NT e pode ser explicado pelo esquema de gerenciamento integrado de *caching* de disco e de memória virtual do Windows NT [4]. A variação na utilização do disco apresentada pelas diferentes cargas é explicada por suas características. A baixa utilização de disco refletida no gráfico da esquerda pode ser explicada pelo fato de que a carga A consistiu de acessos a uma variedade de 18 arquivos, sendo que todos estes arquivos totalizavam 298 KBytes. Como consequência, grande parte destes arquivos poderiam ficar armazenados no *cache* de disco da máquina servidora, poupando vários acessos a disco. Isto não ocorre para a configuração B, porque a soma dos tamanhos dos arquivos que a compõem totaliza 5 MBytes, o que justifica uma atividade mais

intensa de disco e memória que pode ser verificada no gráfico da direita da figura 4. Entretanto, mesmo para esta carga que, segundo [1], é atípica, pode-se verificar que as atividades de disco e memória não chegaram a ser pontos de contenção.

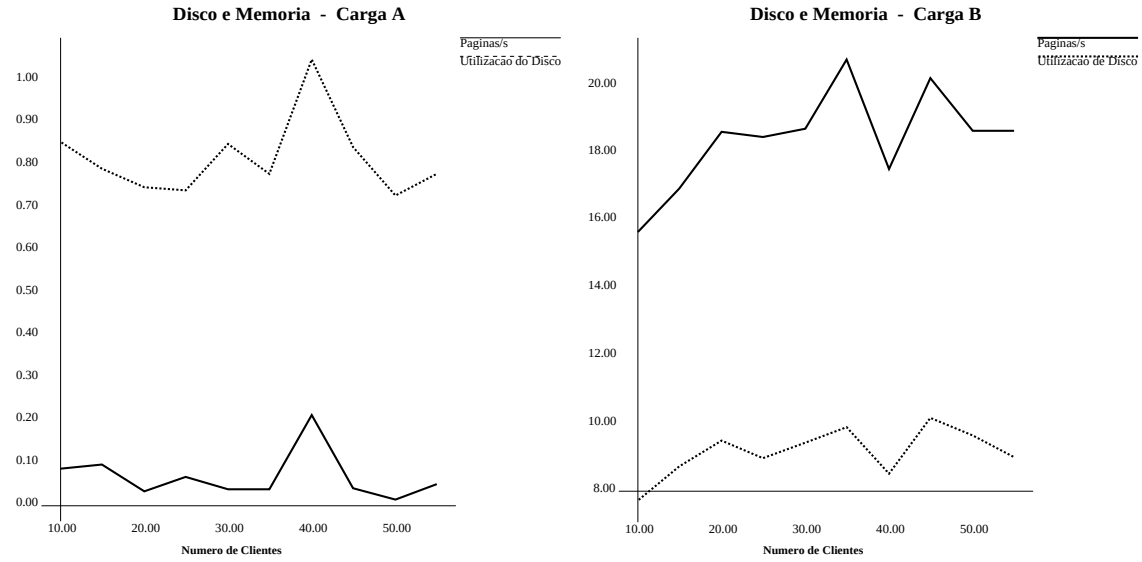

Figura 4: Medidas do Performance Monitor: utilização de disco e de memória

4.4 Little's Load Factor

A figura 5 mostra o comportamento do *Little's Load Factor* em função do número de clientes para as duas cargas. Pode-se observar que as duas curvas coincidem com a curva da função identidade até 25 clientes. A partir deste ponto, as curvas de *Little's Load Factor* mudam de comportamento, os valores crescem mais lentamente indicando que o sistema está aumentando o tempo gasto em atividades que causam *overhead*, como tratamento de interrupções e erros. Isto está em conformidade com os resultados obtidos para utilização do processador pelo *https* no *Performance Monitor*. O *Little's Load Factor* representa o tempo gasto no servidor para processamento das requisições. Os valores são ótimos para até 25 clientes, ponto de máxima utilização do processador pelo *https*. A partir daí, o servidor está sobrecarregado; logo os valores de *Little's Load Factor* tendem a crescer mais lentamente.

5 Conclusões

Este artigo apresenta uma análise do desempenho de um servidor Web no ambiente Pentium/Windows NT. Utilizando uma ferramenta de medição de desempenho fornecida pelo Windows NT e o *benchmark* WebStone, foram feitos vários experimentos para investigar o comportamento e os pontos de contenção do servidor. Os valores máximos obtidos para *throughput* e taxa de conexão foram, respectivamente, 21.58 Gbytes/dia e 1.54 milhão de conexões/dia. Estes valores são grandes se comparados às demandas médias da maioria dos servidores atuais. Um servidor Web da NCSA - *National Computer Security Association* , um dos *sites* mais acessados, atende, em média, 24419 conexões/dia [13]. Além disto, estes valores são de pico, e isto não

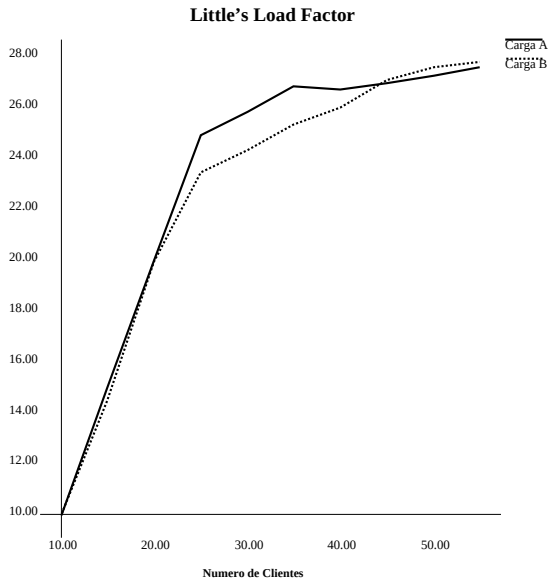

Figura 5: Medidas do WebStone: *Little's Load Factor* para as duas cargas

representa o comportamento típico do acesso à Web. Porém, o objetivo foi analisar o aspecto de saturação de um servidor Web, identificando o número máximo de clientes para o qual o sistema está apto a processar.

O recurso que tem mais impacto no desempenho do servidor é o processador. A partir de 25 clientes (ponto de saturação), o *overhead* do sistema aumenta e o *throughput* diminui. As utilizações de disco e memória não foram significativas a ponto de representarem pontos de contenção.

Este trabalho apresenta alguns aspectos gerais importantes como a necessidade de se monitorar tanto do lado do cliente quanto do lado do servidor, uma vez que este vê seus recursos sendo gastos no processamento de requisições cujas respostas podem não chegar ao cliente. Outro aspecto é a tendência atual de utilização da Web. Instituições que tornam disponíveis informações na Web têm interesse em monitorar o desempenho de seus servidores para verificar a quantidade de erros e o tempo médio de resposta. Um critério que vem sendo utilizado é que as páginas valem pelo número de pessoas que as visitam. Por isto, o tempo de resposta e o número de erros têm que ser minimizados para incentivar os usuários a utilizar este recurso.

Referências

[1] Arlitt, M. F. & Williamson, C. L., "Web Server Workload Characterization: The Search for Invariants", Department of Computer Science, University of Saskatchewan, Canada, October 1995.

[2] Berners-Lee, T., Cailliau, R., Luotonen, A., Nielsen, H. & Secret, A., "The World Wide Web", *Communications of the ACM*, Vol. 37(8), August 1994.

[3] Bestavros, A., Carter, R., Crovella, M., Cunha, C., Heddaya, A. & Mirdad, S., "Application-Level Document Caching in the Internet", *Proceedings of the*

Second International Workshop on Services in Distributed and Networked Environments (SDNE '95)", Whistler Canada, June 1995.

[4] Blake, R., *Optimizing Windows NT*, Microsoft Press, 1993.

[5] Blakeley, M., "WebStone Performance Analysis: Windows NT 3.5", December 1995, URL: http://www-europe.sgi.com/Products/WebFORCE/WebStone/nt-p100/nt-p100.html.

[6] Braun, H. & Claffy, K., "Web Traffic Characterization: An Assessment of the Impact of Caching Documents from NCSA's Web Server", *Electronic Proceedings of the Second World Wide Web Conference '94: Mosaic and the Web*, Chicago, Illinois, October 1994.

[7] Cunha, C., Bestavros, A. & Crovella, M., "Characteristics of WWW Client-Based Traces", Technical Report BU-CS-95-010, Boston University, Computer Science Department, 1995.

[8] European Microsoft Windows NT Academic Centre, HTTP Server Manual Version 0.99,
URL: http://emwac.ed.ac.uk/html/internet_toolchest/https/contents.htm.

[9] Frentzen, J. & Sullivan, E., "Secure Servers", October 1995, URL: http://www.zdnet.com/ pcweek/sr/1030/tserv.html.

[10] Gray, M., "Measuring the Growth of the Web — June 1993 to June 1995",
URL: http://www.netgen.com/info/growth.html.

[11] McGrath, R. E., "Performance of Several HTTP Daemons on an HP 735 Workstation", *Computing & Communications*, NCSA, April 1995.

[12] Menascé, D., Almeida, V. & Dowdy, L., *Capacity Planning and Performance Modeling: from Mainframes to Client/Server Systems*, PTR Prentice-Hall, Englewood Cliffs, 1994.

[13] NCSA Web Server Activity: Total Connections,
URL: http://www.nca.uiucedu/SDG/Presen.stations/Stats/WebServer.html.

[14] Padmanabhan, V. N. & Mogul, J. C., "Improving HTTP Latency", *Electronic Proceedings of the Second World Wide Web Conference '94: Mosaic and the Web*, Chicago, Illinois, October 1994.

[15] The World Wide Web Consortium, URL: http://www.w3.org/pub/WWW/Protocols.

[16] Trent, G. & Sake, M. "WebSTONE: The First Generation in HTTP Server Benchmarking", February 1995,
URL: http://www.sgi.com/Products/WebFORCE/WebStone.