Estratégias de Balanceamento de Carga em Servidores Web Transacionais

T. Tavares G. Teodoro D. Nogueira B. Coutinho
W. Meira Jr. D. Guedes

[1]Departmento de Ciência da Computação
Universidade Federal de Minas Gerais
Belo Horizonte MG Brazil 31270-010
{ttavares,george,diego,coutinho,meira,dorgival}@dcc.ufmg.br

Abstract. *One of the main challenges to the wide use of the Internet is the scalability of the servers, that is, their ability to handle the increasing demand without affecting the quality of the services provided. Scalability in stateful servers, which comprise e-Commerce and other transaction-oriented servers, is even more difficult, since it is necessary to keep transaction data across requests from the same user. One common strategy for achieving scalability is to employ clustered servers, where the load is distributed among the various servers. However, as a consequence of the workload characteristics and the need of maintaining data coherent among the servers that compose the cluster, load imbalance arise among servers, reducing the efficiency of the server as a whole. In this paper we propose and evaluate a strategy for load balancing in stateful clustered servers that takes into consideration workload characteristics. Our strategy is based on control theory and allowed significant gains over configurations that do not employ load balancing strategies or standard load balancing strategies, reducing the response time in up to 18% and increasing the throughput in up to 14%.*

Resumo. *Um dos maiores desafios para o uso amplo da Internet é a escalabilidade dos servidores, ou seja, a sua capacidade em suportar uma demanda crescente sem que a qualidade dos serviços providos seja afetada. Escalabilidade em servidores transacionais, que compreendem servidores de comércio eletrônico e governo eletrônico, é ainda mais difícil, uma vez que é necessário manter dados das transações de um mesmo usuário durante a sua interação. Uma estratégia comum para conseguir escalabilidade é utilizar servidores agrupados, onde a carga é distribuída entre os servidores. Entretanto, como uma conseqüência das características da carga de trabalho e da necessidade da manutenção de dados coerentes entre os servidores agrupados, pode surgir desbalanceamento de carga entre os servidores, reduzindo a sua eficiência. Neste artigo propomos e avaliamos uma estratégia de balanceamento de carga que é função das características da carga de trabalho. Nossa estratégia é baseada em teoria do controle e permitiu ganhos significativos sobre configurações que não empregam balanceamento de carga ou mesmo estratégias tradicionais, reduzindo o tempo de resposta em até 18% e aumentando a taxa de serviço em até 14%.*

1. Introdução

O crescimento recente da Internet está relacionado à expansão da *World Wide Web* (WWW). Uma parte significativa deste crescimento pode ser creditado ao aumento do número de serviços e usuários de comércio eletrônico e outras aplicações transacionais que se utilizam da WWW.

Uma medida frequente de sucesso de aplicações WWW e aplicações de comércio eletrônico em particular é a capacidade do sítio de responder requisições prontamente. No caso de servidores de comércio eletrônico, o sucesso do serviço é conseqüência da capacidade do servidor de capturar a atenção de um grande número de usuários e mantê-los satisfeitos com a qualidade do serviço provido. Por outro lado, um grande número de usuários significa sobrecarga nos servidores responsáveis pelos serviços, a qual não deve afetar a experiência dos clientes.

Em conseqüência, escalabilidade se tornou uma questão fundamental dos servidores de comércio eletrônico. Mas aumentar a capacidade de um servidor nem sempre é possível, pois há limites para a velocidade do processador e da memória, entre outros fatores [Nahum et al., 2002]. Uma estratégia comum nesses casos é o uso de servidores agrupados, onde as capacidades de máquinas individuais não afetam diretamente a qualidade do serviço provido. Essa estratégia tem sido aplicada com sucesso em servidores de conteúdo estático, motivando o desenvolvimento de novas técnicas de distribuição e identificando gargalos nesses sistemas. Uma boa descrição das técnicas de distribuição de servidores Web é apresentada por Cardellini [Cardellini et al., 2002]. Por outro lado, grande parte do trabalho na área de análise de desempenho de servidores de comércio eletrônico tem sido feita no contexto de servidores individuais [Amza et al., 2002, Cecchet et al., 2002]. A distribuição de serviços traz uma nova dimensão para o problema, que não tem sido muito discutida na literatura.

Um desafio enfrentado pelos servidores agrupados, no caso de servidors dinâmicos como os de comércio eletrônico, é que o ganho de desempenho (*speed-up*), isto é, a melhoria do desempenho resultante do aumento do número de processadores, não é linear. A distribuição de requisições entre um número de servidores impõe novas demandas de processamento para garantir a consistência das informações armazenadas por eles. Por exemplo, se dois clientes, sendo atendidos por dois servidores diferentes, tentam adquirir o mesmo produto, os dois servidores devem garantir que o produto não será vendido duas vezes. Esta computação adicional pode crescer a um ponto tal que o acréscimo de servidores pode piorar o desempenho global.

Outra fonte de custos adicionais, além da manutenção da consistência, é o gerenciamento da distribuição de requisições aliado à manutenção de estado. O protocolo HTTP não prevê a manutenção de estado, de tal forma que cada requisição é tratada de forma independente de outras submetidas pelo mesmo usuário, podendo ser enviada a qualquer servidor que compõe o servidor agrupado. Entretanto, em serviços de comércio eletrônico, por exemplo, a interação do cliente com o servidor cria uma certa quantidade de informação de estado, como os produtos adicionados a uma cesta de compras e identificação dos usuários, entre outros. Nesse caso, a distribuição de requisições deve considerar a existência e localização dessa informação de estado, o que torna a tarefa mais complexa. Todas essas características podem resultar em desbalanceamento de carga entre servidores que são responsáveis por atender as requisições.

Estratégias de balanceamento de carga usualmente se baseiam no princípio de que uma redistribuição bem planejada de carga entre os servidores equaliza as cargas dos mesmos. O planejamento das operações de transferência de carga entre servidores por sua vez se baseia no princípio de que a carga migrada mantém a sua intensidade após a migração (é possível prever o comportamento da carga após sua transferência). O que se verifica na prática, entretanto, é que a carga de trabalho de servidores de comércio eletrônico normalmente é caracterizada por uma alta variabilidade em diversos dos seus componentes. Essa variabilidade afeta significativamente a eficácia de estratégias de balanceamento de carga, pois torna o trabalho de prever o comportamento futuro da carga a ser migrada um grande desafio.

Neste artigo analisamos as características de cargas de trabalho de comércio eletrônico reais à luz do seu impacto em termos da eficácia de estratégias de balanceamento de carga. A partir dessa análise, propomos e avaliamos novas estratégias de balanceamento de carga que levam em conta essas características da carga, sendo mais eficazes.

O restante desse documento é organizado da seguinte maneira: na Seção 2 introduzimos a arquitetura e os componentes principais de Servidores Web Transacionais, seguimos com uma discução dos tópicos de balanceamento de carga nesses servidores na Seção 3. Por último, mostraremos os experimentos realizados na Seção 4, e a conclusão na Seção 5.

2. Servidores Web Transacionais

Com o objetivo de entender os detalhes de um servidor Web transacional, é necessário conhecer as relações entre as várias entidades envolvidas e a arquitetura do servidor. Nesta seção discutimos esses aspectos. Sem perda de generalidade, nós podemos discutir os detalhes de servidores transacionais através da análise de uma arquitetura típica: servidores de comércio eletrônico, ou lojas virtuais.

2.1. Entidades Básicas

Há basicamente três entidades básicas em serviços de comércio eletrônico: produtos, clientes e sessões.

A primeira entidade compreende os objetos das transações, havendo dois tipos de dados a respeito deles: estático e dinâmico. Dados estáticos modelam informações que não são afetadas pelos serviços providos pelo servidor, como a descrição de um livro, ou as características de um equipamento eletrônico. Dados dinâmicos, por outro lado, modelam informações que são alteradas pelas operações executadas durante o provimento dos serviços, como o estoque dos produtos, ou a identidade do comprador.

A segunda entidade representa os clientes, que também demandam o armazenamento de dados estáticos e dinâmicos. Neste caso, informações como nome e endereço são estáticas, enquanto outros atributos, como o conteúdo da cesta de compras, são dinâmicos.

A identificação de dados estáticos e dinâmicos é crucial para a distribuição dos serviços em servidores agrupados. Enquanto dados estáticos podem ser facilmente replicados nos servi-

dores agrupados, acessos a dados dinâmicos devem ser controlados de forma a evitar inconsistências.

A terceira entidade representa a relação entre as duas primeiras entidades, isto é, a interação entre clientes e produtos. Essa interação é sintetizada pela sessão de usuário no contexto de servidores de comércio eletrônico, a qual é construída à medida que o servidor responde às requisições do usuário. Uma sessão de servidor de comércio eletrônico combina referências a dados tanto estáticos quanto dinâmicos de clientes e produtos que estejam associados à sua interação.

2.2. Organização em Níveis

Para armazenamento de dados e atendimento às requisições, os servidores transacionais são estruturados como uma arquitetura de três níveis [Wrigley, 1997, McDavid, 1999]: servidor WWW, servidor de aplicações e servidor de banco de dados. Esses níveis são ilustrados na Figura 1 e discutidos a seguir.

1. **Servidor WWW**: É o gerente das tarefas, sendo responsável pela interface com os usuários (clientes), interagindo diretamente com eles, recebendo as solicitações de consultas ao banco de dados, disparando-as, e repassando os resultados. Provê a interface entre a ferramenta de acesso do cliente (normalmente um *browser*) e o servidor de comércio eletrônico.
2. **Servidor de Aplicação**: Realiza o processamento das requisições submetidas ao servidor de comércio eletrônico, como a adição/remoção de um novo produto à cesta de compras. Implementa a lógica específica do negócio.
3. **Banco de Dados**: Armazena todas as informações da loja virtual, como descrição do produto e nível de estoque. Mais que um simples repositório, agrega uma série de funcionalidades que permitem o acesso padronizado, seguro e eficiente aos dados, através, por exemplo, da criação de índices e controle de acesso utilizando autenticação por usuário.

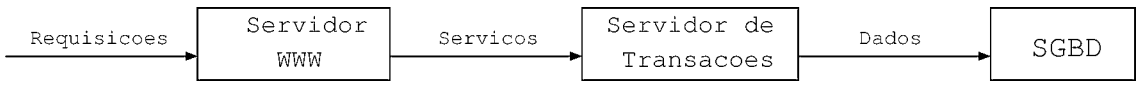

Figura 1: Arquitetura de servidor de comércio eletrônico

Em servidores de comércio eletrônico centralizados, os três níveis podem ser implementados como uma aplicação monolítica por razões de desempenho. Entretanto, na maioria dos casos, o serviço é implementado utilizando processos separados para cada um dos componentes. Uma primeira abordagem para melhorar o desempenho é simplesmente usar máquinas separadas para cada um dos componentes, o que tem um alcance limitado, pois para que tenhamos escalabilidade efetiva é necessário que componentes de processamento sejam replicados e as requisições distribuídas adequadamente. No caso de serem usados servidores agrupados na implementação de serviços transacionais, esses são normalmente configurados como vários servidores WWW e de aplicação, com um só servidor de banco de dados. É possível utilizar parte

do agrupamento para se implementar um servidor de banco de dados distribuído, que é uma tecnologia mais complexa e dispendiosa que os bancos de dados tradicionais, porém essa ainda não é uma solução viável na maioria dos casos. Em geral, dados não são armazenados no servidor WWW, sendo divididos entre servidores de aplicação e banco de dados. Armazenar dados no servidor de banco de dados é uma solução mais robusta, mas associada a um maior custo, tanto em termos de recursos necessários, quanto em termos de latência de acesso. Por esse motivo, é interessante notar que, apesar do armazenamento persistente ser realizado no servidor de banco de dados, pode ser interessante armazenar ou replicar dados nos servidores de aplicação.

2.3. Distribuição de Dados

Distribuir dados estáticos pode ser visto como um problema de replicação. Cada servidor tem uma capacidade limitada de armazenamento e deve gerenciar esse espaço levando em consideração aspectos como custos de comunicação entre servidores, localidade de referência e custos de armazenamento no servidor. Considerando esses fatores, há várias estratégias de gerenciamento de replicação que podem ser aplicadas. Se o custo de armazenar é baixo, replicação total em todos os servidores de aplicação pode reduzir as latências de resposta, uma vez que todos os servidores vão conter todos os dados. Por outro lado, se o custo de armazenamento é alto, caches mutuamente exclusivos permitem um melhor aproveitamento do espaço, demandando, entretanto, significativa comunicação entre servidores. Algumas soluções intermediárias podem ser adotadas, replicando informações populares e reduzindo a quantidade de comunicação em alguns casos [Pierre et al., 2000].

Distribuir dados dinâmicos é bem mais complicado por questões de coerência, ou seja, se um dado dinâmico é armazenado em dois servidores, deve haver uma forma de garantir que ambos os servidores serão notificados das mudanças. Este problema vem sendo pesquisado no contexto de banco de dados distribuídos. Entretanto, é interessante notar que grande parte da carga de trabalho associada a clientes envolvem dados estáticos [Meira Jr. et al., 2000], como descrição de produtos e imagens, o que representa uma oportunidade para replicação [Gadde et al., 2001].

Considerando os diversos fatores apresentados, neste artigo apresentamos uma estratégia de balanceamento de carga para uma abordagem comum para a construção de servidores de comércio eletrônico: replicação de dados estáticos e não replicação de dados dinâmicos. Esta abordagem é simples em termos de implementação e não demanda mecanismos de cooperação, mas tende a ser pouco escalável para um grande número de servidores. Por outro lado, a não replicação de dados dinâmicos pode resultar em desbalanceamento de carga como conseqüência das características da carga de trabalho e pelo fato de um servidor ser o responsável pelas respostas associadas a uma sessão, independente da demanda computacional que ela representa.

3. Estratégias de Balanceamento de Carga

Nesta seção discutimos o problema do balanceamento de carga em servidores Web transacionais agrupados. Mais especificamente, nós enfocamos a questão de como características da carga de trabalho a que esses servidores estão submetidos podem afetar a efetividade das estratégias de balanceamento de carga.

O ponto de partida dessa discussão são as abordagens tradicionais de balanceamento de carga e os requisitos que essas abordagens impõem. Balanceamento de carga em servidores distribuídos é um processo contínuo e iterativo que é normalmente dividido em três fases: (1) monitoração da carga dos servidores durante um período λ; (2) determinação da quantidade de carga que deve ser migrada entre servidores, com base nas observações de carga; e (3) migração da carga. Um compromisso sempre presente nessas estratégias é com relação ao valor de λ. Se λ é pequeno, o custo de balanceamento (*overhead*) pode se tornar alto e não haver tempo suficiente para que o balanceamento surta efeito. Por outro lado, se λ é grande, o desbalanceamento pode perdurar o bastante a ponto de afetar o desempenho global dos servidores agrupados. Desta forma, determinar o melhor valor de λ é uma operação de calibragem que depende das características do sistema (qual o custo de migrar carga) e da aplicação (quão variável é a carga ao longo do tempo).

Uma vez que o valor de λ tenha sido escolhido e o processo de monitoração de carga esteja em operação é possível determinar o nível de desbalanceamento de cada servidor de um agrupamento. Essa medida pode ser feita, por exemplo, calculando-se a carga média entre todos os servidores e definindo-se o nível de desbalanceamento de cada servidor como sendo a diferença de sua carga em relação à média. Dessa forma podemos identificar servidores sobrecarregados e sub-utilizados no sistema. Uma vez feito isso, não basta que se faça um casamento direto de servidores baseado em suas diferenças de carga, pois essa solução pode levar a instabilidades [Sontag, 1997].

A solução adotada neste trabalho é considerar o agrupamento de servidores como um sistema de controle com realimentação como ilustrado na Figura 2. Um sistema de controle desse tipo opera da seguinte maneira: o usuário define o comportamento esperado para o sistema em termos de uma grandeza mensurável por ele produzida. Esse valor esperado é o valor de referência para o sistema (ou *set point*, SP) — por exemplo, a latência máxima aceitável para requisições dos clientes. O valor real dessa grandeza do sistema é monitorado continuamente (a variável do processo, PV) e o erro do valor real em relação ao valor de referência é calculado, $e = SP - PV$. Com base nesse erro, o controlador do sistema deve calcular um valor de atuação (saída do controlador, CO) que, aplicado ao sistema que se deseja controlar, minimize o erro e levando a variável do processo para o valor definido como referência (SP).

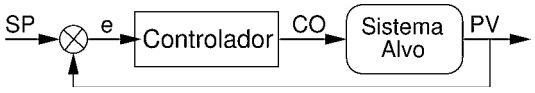

Figura 2: Sistema de controle com realimentação

Do ponto de vista da operação do sistema agregado, a aplciação dessa técnica de controle significa que o desenvolvedor deverá determinar os níveis de desempenho aceitáveis para o sistema (em termos de latência e taxa de atendimento, por exemplo). O controlador então se encarregará de encontrar o padrão de distribuição de carga entre os servidores que garantirá a manutenção daqueles níveis.

Esse enfoque de sistemas de controle com realimentação é usado na teoria de controle

quando a variável mensurável do processo (PV) não possui uma relação clara e direta com a grandeza de atuação usada como entrada do sistema (CO). Nesses casos, o comportamento interno do controlador é responsável por ajustar sua saída dinamicamente, com base no laço de realimentação, até que o valor desejado para a variável de processo seja alcançado. A forma tradicional de se alcançar esse objetivo é implementar o que é chamado um controlador Proporcional-Integral-Derivado (PID) [Rugh, 1996]. Nesse caso, a saída do controlador e o erro medido com relação ao valor de referência são relacionados pela equação:

$$CO = K_p e + K_i \int_0^t e.dt + K_d \frac{de}{dt} + \text{offset}$$

As contantes PID K_p, K_i, K_d definem os pesos das três componentes do erro e devem ser definidas por um processo de sintonia do controlador para cada agrupamento de servidores.

Uma vez determinadas as cargas de cada servidor e a carga média do sistema, cada servidor deve determinar em quanto sua carga deve ser alterada com base na aplicação do controlador PID a fim de buscar o cancelamento do erro, o que implicaria no perfeito balanceamento do sistema. Uma vez determinadas as alterações esperadas de carga, um emparelhamento dos nós pode ser feito combinando nós sobrecarregados, que devem ceder parte da sua carga, com outros sub-utilizados, que devem receber mais carga. Isso pode ser feito, por exemplo, através de uma heurística gulosa. A próxima fase consiste na determinação exata de quais componentes da carga deverão ser transferidos entre servidores.

No contexto de servidores Web transacionais, a terceira fase é bastante complexa, pois nem sempre é trivial selecionar os componentes de carga a serem migrados. No caso de servidores de comércio eletrônico, por exemplo, migrar as sessões de usuários é uma forma de promover o balanceamento de carga entre os servidores.

Migração de carga pode ser visto como um exercício de estimativa, no sentido de que nos baseamos na premissa de que o nível de carga associado ao componente migrado vai se manter e ser assumido pelo servidor destino. O sucesso da migração de carga e portanto da estratégia de balanceamento de carga depende da representatividade da métrica de carga e da habilidade em migrar apenas a carga necessária entre servidores. Como discutimos a seguir, a precisão nesse processo de migração de carga é um desafio.

Nós distinguimos quatro características da carga de trabalho de servidores Web transacionais que são relevantes para o sucesso de estratégias de migração de carga: processo de chegada de clientes, duração da sessão dos clientes, processo de chegada das requisições de uma sessão e custo das requisições. A seguir discutimos cada uma dessas características.

O processo de chegada de clientes em servidores Web transacionais é caracterizado pela sua alta variabilidade. O processo de chegada de clientes em servidores de comércio eletrônico foi modelado por uma distribuição de cauda pesada em [Menascé et al., 2003]. Distribuições de cauda pesada são caracterizadas pela ocorrência de eventos pouco frequentes, mas de grande impacto, como uma rajada de sessões se iniciando em um curto período de tempo. Entretanto, estes fenômenos não afetam diretamente as estratégias de balanceamento de carga, uma vez que pouco se sabe sobre a carga associada a uma sessão quando da sua chegada e mecanismos de

distribuição de carga tradicionais [Cardellini et al., 2002] têm se mostrado bastante efetivos para assinalar novas sessões a servidores de forma balanceada.

A segunda característica é a duração da sessão dos usuários que indica por quanto tempo o servidor deve gerir a sessão. O gráfico da Figura 3 mostra a distribuição de probabilidades da duração das sessões em uma carga de trabalho de comércio eletrônico. Podemos observar que a duração de uma sessão varia significativamente, sendo difícil prever por quanto tempo uma dada sessão gerará carga *a priori*. Em termos de impacto sobre estratégias de balanceamento de carga, a duração da sessão realmente não determina a carga associada a uma dada sessão, mas é condição fundamental para a efetividade de qualquer estratégia, uma vez que se a sessão não persistir pelo menos por um período λ após a sua migração, a eficácia do balanceamento será comprometida. Entretanto, para fins de balanceamento, é importante sabermos a probabilidade de uma sessão permanecer ativa durante o próximo período λ após a migração ter ocorrido. O gráfico da Figura 4 mostra a probabilidade de uma sessão continuar ativa por um período adicional λ, dado que ela já está ativa pelo tempo indicado na abscissa do gráfico. É interessante observar que para a grande maioria das sessões essa probabilidade independe do tempo de atividade das sessões, mas se apresenta como função de λ, ou seja, a probabilidade decresce à medida que o valor de λ aumenta, como esperado. Em conseqüência, o compromisso em torno do valor de λ se torna ainda mais complexo, pois uma redução do valor de λ tende a tornar o balanceamento de carga mais efetivo, mas aumenta os custos adicionais de balanceamento.

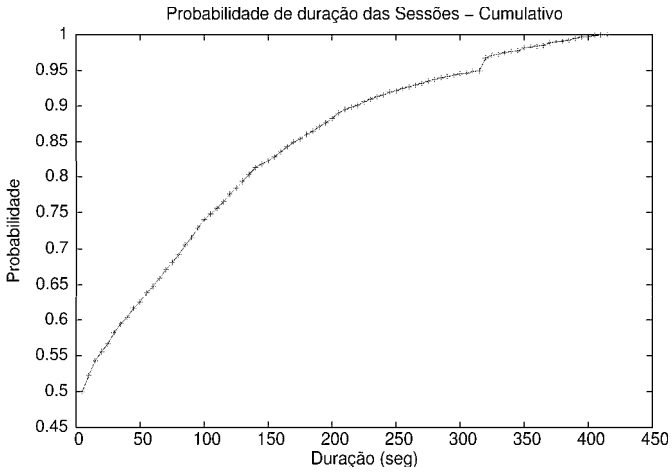

Figura 3: Distribuição da duração das sessões

A terceira característica é o processo de chegada das requisições de uma sessão. Trabalhos anteriores [Menascé et al., 2003] já mostraram que há uma alta variabilidade nos intervalos entre chegadas de requisições das sessões de uma carga de trabalho de um servidor de comércio eletrônico. Mais ainda, essa variabilidade também pode ser observada entre requisições de uma mesma sessão. No contexto de balanceamento de carga, essa variabilidade implica em baixa capacidade de prever a carga associada a uma sessão, comprometendo a efetividade da estratégia de balanceamento de carga.

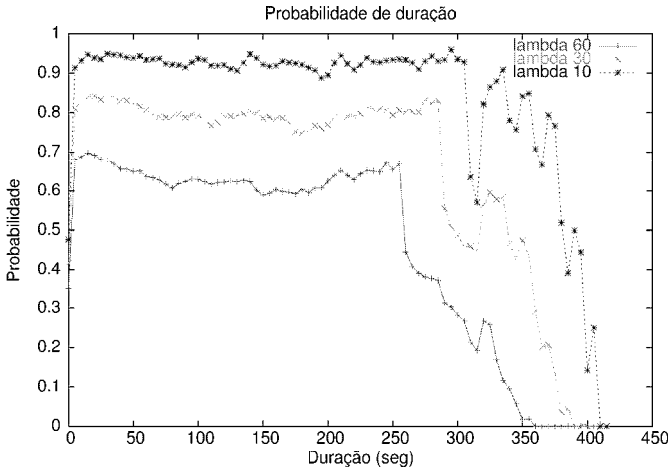

Figura 4: Probabilidade de uma sessão continuar ativa por um período lambda

A última característica é o custo das requisições que, como mencionado, é bastante variável, tanto entre requisições diferentes quanto entre requisições do mesmo tipo, de acordo com a natureza dos seus parâmetros. Do ponto de vista da estratégia de balanceamento de carga, quanto maior a variabilidade das requisições associadas a uma sessão, mais complexa é a tarefa de balanceamento de carga.

Estas várias características indicam que a alta variabilidade observada pode comprometer a efetividade das estratégias de balanceamento, que em geral têm uma preocupação maior em termos de equalizar a carga entre os servidores tanto quanto possível, quando, para servidores transacionais na Web, não está claro se isso é possível. A nossa proposta é uma estratégia de balanceamento de carga que considere a variabilidade das características. No caso específico de servidores de comércio eletrônico, introduzimos o conceito de regularidade da sessão do usuário. Considerando que a migração da carga ocorre periodicamente, a regularidade de uma sessão pode ser definida de forma discreta e quantificada pela variabilidade dos custos das requisições e dos intervalos entre chegadas de requisições. No contexto de políticas de migração de carga, a proposta é que sejam migradas sempre sessões que são mais regulares, aumentando a provável efetividade da migração.

4. Avaliação

Nesta seção apresentamos os resultados simulados e experimentais da estratégia de balanceamento de carga proposta. As próximas duas sub-seções apresentam o simulador e o ambiente experimental, seguido dos resultados. Simulação foi empregada com o objetivo de quantificar o impacto de variações na carga de trabalho, enquanto os resultados experimentais avaliam a eficiência das estratégias em uma execução real.

4.1. Simulador

O simulador foi implementado em linguagem Java e segue uma arquitetura tradicional de simulação baseada em eventos. Em particular, o simulador modela um grupo de servidores de aplicação e as suas interações ao atender as requisições, quantificando a contenção por processamento nos servidores de aplicação. Para tal, o simulador considera as latências de comunicação e de acesso ao servidor de banco de dados.

Estão fora do escopo do simulador a latência da Internet e os custos do servidor WWW. Uma outra premissa é que o servidor de banco de dados não possui situações de contenção. A entrada do simulador é um arquivo de eventos contendo a sua natureza (início ou fim de requisições, assim como início ou fim de escopos de processamento), duração e tempo de início.

Em termos de estratégias de balanceamento de carga, o simulador suporta três estratégias diferentes: (1) sem balanceamento, (2) balanceamento baseado em carga das sessões e (3) balanceamento baseado em regularidade.

Os experimentos apresentados na Seção 4.3 têm por objetivo avaliar as estratégias de balanceamento de carga como função das características da carga de trabalho.

4.2. Ambiente Experimental

Nesta seção discutimos a instalação experimental usada para avaliar a estratégia de balanceamento proposta.

Distinguimos quatro componentes integrados no servidor de transações utilizado nos experimentos: (1) servidor Web (Apache 1.3.20 [Foundation, 1999]), (2) servidor de aplicações paralelo implementado pelos autores, e (3) sistema de gerenciamento de banco de dados (MySQL 3.23.51), e o diretório de sessões, que é responsável pelo controle de quais sessões são geridas por quais servidores.

A carga de trabalho é submetida por clientes Httperf [Mosberger and Jin, 1998] que modificamos de tal forma que as requisições sejam redirecionadas sempre que houver uma migração de carga.

Os resultados foram obtidos em um grupo de 10 máquinas executando Linux 2.4, sendo seis AMD Duron 750Mhz com 256 Mb RAM e quatro AMD Athlon 750Mhz com 256Mb RAM. Essas máquinas se comunicam através de uma chave Fast Ethernet e atuam da seguinte forma. As quatro máquinas Athlon geram carga. Um Duron atua como servidor de banco de dados e outro como servidor de diretório. As outras quatro máquinas executam cada, um servidor Web e um servidor de aplicações, opção justificada pela carga usualmente baixa do servidor Web e da redução da latência entre esses dois servidores.

As cargas de trabalho geradas por cada cliente são baseadas na carga de trabalho caracterizada em [Menascé et al., 2003], sendo as sessões igualmente distribuídas entre os clientes em uma estratégia rotativa. Com o objetivo de avaliar a efetividade das cargas de trabalho, as cargas são deliberadamente desbalanceadas, ou seja, uma fração significativa das requisições é submetida a um único servidor.

Estratégia	Taxa de Serviço (req/s)	Tempo de Resposta (s)
Sem balanceamento	44.6475	0.2981
Balanceamento - Carga	44.4965	0.2834
Balanceamento - Regularidade	51.0495	0.2433

Tabela 1: Resultados Experimentais - 4 servidores

4.3. Resultados

Nesta seção apresentamos os resultados de simulação e os resultados experimentais obtidos com a estratégia de balanceamento de carga proposta.

Um primeiro aspecto avaliado na simulação é com relação ao impacto da variabilidade do processo de chegada das requisições na eficácia da estratégia de balanceamento de carga. Para que pudéssemos simular as situações desejadas, alteramos os logs utilizados na simulação para refletir uma maior variabilidade naquele processo de chegada. Isso foi atingido através da substituição dos tempos entre requisições por outros, que foram gerados seguindo uma distribuição de Pareto com coeficientes variando de 0.5 a 4.5. Os resultados mostraram que, conforme esperado, a variabilidade do processo de chegada torna a tarefa de balaceamento mais complicada, mas, entretanto, a estratégia de balanceamento proposta reduziu em aproximadamente 10% o desbalanceamento médio, quando comparada com a estratégia de balanceamento que busca apenas equalizar carga. Esses resultados de simulação anteciparam os resultados experimentais.

A seguir apresentamos resultados experimentais da execução de um servidor de comércio eletrônico agrupado. Avaliamos três configurações: sem balanceamento, com balanceamento baseado apenas em redistribuição de carga e com balanceamento considerando a regularidade das sessões. Os experimentos consistiram de submeter uma carga de trabalho completamente desbalanceada, isto é, todas as requisições são encaminhadas a um único servidor a menos que sessões sejam migradas. Os resultados são mostrados na Tabela 1 onde podemos verificar que a estratégia baseada em regularidade foi significativamente melhor tanto em termos de taxa de serviço (18%) quanto tempo de resposta (14%).

Este resultado pode ser explicado se avaliarmos o gráfico da Figura 5, onde é apresentado o desbalanceamento médio dos servidores ao longo do experimento. Definimos desbalanceamento como a diferença entre a carga observada no servidor e a carga esperada (a carga total dividida pelo número de servidores). Nesse caso, podemos perceber que a estratégia baseada em regularidade sempre permitiu um melhor resultado, a menos de pequenas oscilações.

5. Conclusão e Trabalhos Futuros

Neste artigo apresentamos uma análise do problema de balanceamento de carga em servidores transacionais em função das características da carga de trabalho desse tipo de sistema e propusemos uma nova estratégia de balanceamento que leva em conta essas características.

Para validar a estratégia proposta avaliamos um servidor de comércio eletrônico agrupado em três configurações: sem balanceamento, com balanceamento baseado apenas na redistribuição

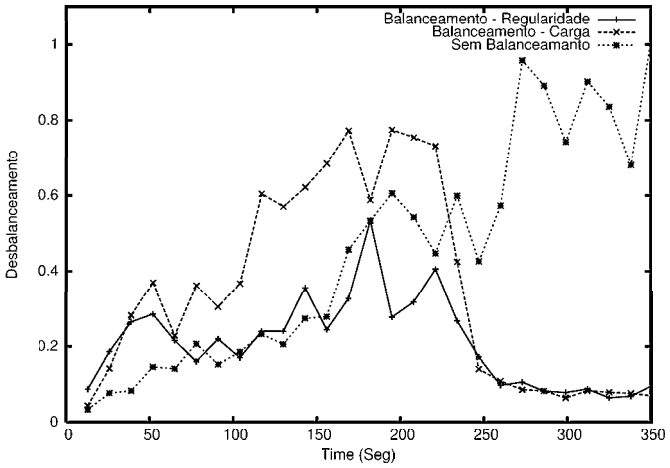

Figura 5: Média de desbalanceamento entre os Servidores

de carga e com balanceamento considerando a regularidade das sessões. A avaliação envolveu o uso de um modelo simulado e sua validação com base na utilização de uma loja virtual implementada em nosso laboratório.

A implementação da estratégia proposta é baseada em teoria de controle e permitiu ganhos significativos sobre configurações que não empregam balanceamento de carga ou mesmo estratégias tradicionais de balanceamento, reduzindo o tempo de resposta em até 18% e aumentando a taxa de serviço em até 14%.

Como trabalhos futuros pretendemos continuar as análises de sensibilidade das políticas em função de variações das características da carga, aprimorando o simulador para incluir mais detalhes do sistema. Além disso pretendemos continuar com o processo de validação dos resultados utilizando a loja virtual com carga real.

Referências

Amza, C., Cecchet, E., Chanda, A., Cox, A., Elnikety, S., Gil, R., Marguerite, J., Rajamani, K., and Zwaenepoel, W. (2002). Bottleneck characterization of dynamic web site benchmarks. In *Third IBM CAS Conference*. IBM.

Cardellini, V., Casalicchio, E., Colajanni, M., and Yu, P. (2002). The state of the art in locally distributed web-server systems. *ACM Computing Surveys*, 34(2):1–49.

Cecchet, E., Chanda, A., Elnikety, S., Marguerite, J., and Zwaenepoel, W. (2002). A comparison of software architectures for e-business applications. Technical Report TR02-389, Rice University.

Foundation, T. A. S. (1999).
http://www.apache.org/.

Gadde, S., Chase, J. S., and Rabinovich, M. (2001). Web caching and content distribution: a view from the interior. *Computer Communications*, 24(2):222–231.

McDavid, D. (1999). A standard for business architecture description. *IBM Systems Journal*, 38(1).

Meira Jr., W., Menascé, D., Almeida, V., and Fonseca, R. (2000). E-representative: a scalability scheme for e-commerce. In *Proceedings of the Second International Workshop on Advanced Issues of E-Commerce and Web-based Information Systems (WECWIS'00)*, pages 168–175.

Menascé, D., Almeida, V., Riedi, R., Peligrinelli, F., Fonseca, R., and Jr., W. M. (2003). A hierarchical and multiscale approach to analyze e-business workloads. *Performance Evaluation*, 54(1):33–57.

Mosberger, D. and Jin, T. (1998). httperf: A tool for measuring web server performance. In *First Workshop on Internet Server Performance*, pages 59—67. ACM.

Nahum, E., Barzilai, T., and Kandlur, D. D. (2002). Performance issues in www servers. *IEEE/ACM Transactions on Networking (TON)*, 10(1):2–11.

Pierre, G., Kuz, I., van Steen, M., and Tanenbaum, A. S. (2000). Differentiated strategies for replicating Web documents. In *Proceedings of the 5th International Web Caching and Content Delivery Workshop*.

Rugh, W. J. (1996). *Linear System Theory*. Prentice Hall, second edition edition.

Sontag, E. (1997). A notion of input to output stability. European Control Conf. ECC97.

Wrigley, C. (1997). Design criteria for electronic market servers. *Electronic Markets*, 7(4).