ERNESTO CLEYTON LINHARES DA SILVA

Análise da Interdependência de Métricas de Desempenho na Composição de Clusters de Servidores Web

Dissertação apresentada como requisito parcial para a obtenção do título de Mestre em Informática no Programa de Pós-Graduação em Informática, Setor de Ciências Exatas, Universidade Federal do Paraná.
Orientadora: Profa. Dra. Cristina Duarte Murta

CURITIBA
2004

Ministério da Educação
Universidade Federal do Paraná
Programa de Pós-graduação em Informática

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DO PARANÁ
Setor de Ciências Exatas
Coordenação de Pós-Graduação em Informática

PARECER

Nós, abaixo assinados, membros da Banca Examinadora da defesa de Dissertação de Mestrado em Informática, do aluno *Ernesto Cleyton Linhares da Silva*, avaliamos o trabalho intitulado "*ANÁLISE DA INTERDEPENDÊNCIA DE MÉTRICAS DE DESEMPENHO NA COMPOSIÇÃO DE CLUSTERS DE SERVIDORES WEB*", cuja defesa foi realizada no dia 27 de janeiro de 2005, às treze horas e trinta minutos, no Auditório do Departamento de Informática do Setor de Ciencias Exatas da Universidade Federal do Paraná. Após a avaliação, decidimos pela aprovação do candidato.

Curitiba, 27 de janeiro de 2005.

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

Prof. Dr. Wagner Meira Júnior
DCC/UFMG – Membro Externo

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

AGRADECIMENTOS

Agradeço primeiramente a Deus, que me deu a oportunidade de vir para Curitiba para cursar este mestrado. Em segundo lugar agradeço a minha orientadora Cristina, que teve paciência e dedicação para orientar esta dissertação.

Agradeço aos meus pais, que prestaram todo o suporte que eu precisei para me manter aqui em Curitiba, além de sempre me motivar com palavras de força e incentivo. Agradeço a minha irmã, que veio ficar comigo aqui e me apoiar nas horas de dificuldades.

Agradeço também a Cátia, que muito me ajudou durante todo o curso, seja tirando dúvidas ou emprestando os materiais que precisava.

Por fim, agradeço a todos aqueles que direta ou indiretamente contribuíram para que este trabalho fosse realizado.

Ernesto Linhares

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

RESUMO

A composição de clusters de servidores Web requer a análise de parâmetros de desempenho, confiabilidade e custo. No entanto, estes parâmetros são interdependentes, e relação de dependência entre eles deve ser bem entendida para que os objetivos pretendidos com o sistema sejam atendidos. Esta dissertação apresenta e discute as relações de interdependência entre as principais métricas de um cluster e as modificações que ocorrem nestas métricas quando a composição do cluster é alterada. Alguns exemplos de clusters são analisados e comparados, entre eles um exemplo baseado no cluster do Google. Os resultados indicam que existem grandes variações na composição dos clusters e nos valores das demais métricas de acordo com a métrica selecionada para o estudo de caso.

ABSTRACT

The composition of clusters of Web servers requires the analysis of parameters of performance, reliability and cost. However, these parameters are interdependent, and relation of dependence between them must be understood so that the system's goals are met. This dissertation presents and discusses the relationships observed between the main metrics in a Web cluster as a function of its parameters. Some examples of clusters are analyzed and compared, including an example based on the cluster of Google. The results indicate huge variations in the composition of the clusters and in the measured metrics according to the metric selected for the case study.

CAPÍTULO 1
INTRODUÇÃO

A Internet tem se tornado um recurso de grande importância para muitas pessoas, empresas e instituições. Correio eletrônico, pesquisas, troca de informações são recursos indispensáveis para pessoas e empresas atualmente. Aplicações Web são cada vez mais comuns e são responsáveis pelo aumento da produtividade e redução de custos.

Nos últimos anos foi notável o aumento no fluxo de informações trafegando pela Web, aumentando em complexidade e escala o que é transmitido pela rede. Este aumento no tráfego ocorreu devido a popularização da Web através da disponibilização de uma interface gráfica amigável, a funcionalidade provida pelo hipertexto e a capacidade de publicação de documentos HTML [30]. Este aumento de fluxo acarretou o aparecimento de diversos problemas, onde entre os mais perceptíveis foram a lentidão de acesso e as falhas de conexão, principalmente quando conexões entre continentes são realizadas [29,37]. Nas condições em que a carga trabalhada é alta, os servidores Web precisam servir milhares de requisições concorrentemente [8, 11]. O desempenho do servidor é de grande importância para o desempenho global da Web [2]. Sendo assim, com este crescimento de carga na Internet é necessário otimizar os acessos aos servidores.

Diversos métodos para melhorar os servidores estão sendo estudados, como mostrado em [38]. Existem casos em que sites da Web que precisam processar um grande volume de dados são organizados em grupos para conseguir responder a todas as requisições sem erros para os clientes. Estas organizações de múltiplos servidores e suas vantagens estão sendo amplamente analisadas. Uma mostra destes estudos pode ser vista em [6, 33]. Os agrupamentos de servidores organizados pode ser classificados em duas categorias principais: servidores distribuídos e agrupamento de servidores.

Um grupo de servidores distribuídos é um conjunto de computadores interligados por uma rede de alta velocidade, onde cada computador possui seu próprio endereço IP e seus nodos são visíveis às aplicações-clientes [6].

Um agrupamento de servidores é um conjunto de computadores interconectados por meio de uma rede de alta velocidade, de modo a apresentar para seus usuários a imagem de um sistema único [6,36]. Isto significa que os usuários não conseguem identificar se existe um ou mais computadores no sistema. Este agrupamento de servidores também pode ser chamado de cluster, nomenclatura utilizada neste trabalho. Assim, um cluster é, para quem o usa, indistinguível de um único computador. Cada nodo servidor do cluster é, normalmente, um computador comum, tal como um PC, com disco e uma instalação completa do sistema operacional. Os nodos do cluster operam coletivamente como um único recurso. Um sistema Web baseado em cluster publica um nome único para o site, bem como um único endereço IP. Um servidor Web de uma empresa ou instituição, por exemplo, pode ser composto por uma única máquina ou por dezenas de máquinas. Um dos clusters mais famosos é o do Google [17, 27].

A composição de um cluster abrange questões complexas de pesquisa em computação tais como escalabilidade, disponibilidade, tolerância à falhas, desempenho, e a relação destes fatores com o custo.

Para montar a composição de um cluster, todos estes fatores devem ser levados em consideração, além de métricas como tempo de resposta, capacidade e confiabilidade. Porém existe uma relação entre as métricas de desempenho que alteram a formação do cluster e conseqüentemente seu desempenho, podendo fazer com que este cluster não satisfaça as condições necessárias para executar uma determinada tarefa. Por este motivo, é necessário estudar esta relação de interdependência para saber como estas métricas interagem entre si. É importante lembrar que a metodologia empregada nesta dissertação necessita que uma métrica seja escolhida como padrão para que o compromisso da composição dos clusters possa ser alcançado, seguindo um dos casos analisados. Esta escolha pode ser feita de acordo com as necessidades de cada usuário.

Esta dissertação trata então da análise das relações recíprocas de dependência entre métricas de desempenho e de custo de um cluster. A metodologia utilizada nesta comparação foi descrita em [20] e é baseada nas leis operacionais [17] e em modelos analíticos da teoria de filas [1]. No entanto, o trabalho [20] apresenta valores de comparação um tanto irreais, como servidores Web que servem apenas 20 requisições por segundo. A dissertação estende o trabalho anterior em três aspectos. Em primeiro lugar, utilizamos dados de sistemas

reais, o que torna nossa comparação muito mais realística e plausível. Os sistemas apresentados aqui podem ser comumente encontrados na prática. Em segundo lugar, apresentamos também uma comparação das métricas com valores de parâmetros baseados num grande cluster, o cluster do Google. Finalmente, nós apresentamos resultados para uma métrica essencial neste tipo de análise, que é a razão custo/desempenho.

1.1 Importância e Objetivos

Grandes sites, como os de comércio eletrônico, por exemplo, são compostos por uma arquitetura que possui um ou mais balanceadores de carga, servidores Web, servidores de aplicação e servidores de banco de dados. Estes servidores trabalham em conjunto e normalmente são organizados em clusters por tipo de utilização, onde todos trocam informações entre si. O objetivo principal dos clusters é prover aumento na confiabilidade, diminuição no tempo de resposta e melhorias na capacidade de expansão do sistema. Estes clusters podem ser compostos por diversos servidores com baixo poder de processamento ou o inverso, onde poucos servidores com alto poder de processamento estão presentes.

Muitos sites utilizam diversos servidores com capacidade pequena enquanto outros preferem menos servidores, mas com capacidade individual de processamento maior. Por este motivo, é necessário fazer um estudo para se saber qual é a melhor metodologia a ser empregada em cada caso. Sendo assim, é preciso analisar cada situação em relação a diversas métricas de desempenho, levando-se em consideração qual delas é a mais relevante para a empresa que irá utilizar o cluster. Dentre estas métricas, podemos destacar o tempo de resposta, confiabilidade e custo de projeto.

Além disso, é necessário estudar a utilização de balanceadores de carga em conjunto com os clusters. O balanceador é o dispositivo que tem o trabalho de distribuir a carga para os servidores do cluster seguindo algum critério pré-determinado. Alguns balanceadores de carga podem distribuir a carga igualitariamente ou de acordo com a capacidade individual de cada servidor individualmente, por exemplo. Podem também ser utilizado um grupo de balanceadores, principalmente no caso de grande número de servidores em um ou mais clusters. O balanceamento da carga não faz parte dos objetivos deste trabalho.

Sendo assim, neste estudo assumiu-se que o balanceamento da carga é perfeito, ou seja, todos os servidores recebem a mesma carga para ser executada.

1.2 Organização do Texto

Esta dissertação está organizada da seguinte forma. O capítulo 2 apresenta conceitos e trabalhos relacionados. O capítulo 3 apresenta os modelos analíticos e ilustra sua aplicação com um modelo de cluster pequeno, de 15 máquinas, cada uma com capacidade para servir 100 requisições por segundo. Este cluster é comparado com outro cluster composto por um conjunto menor de máquinas mais potentes, onde o número de máquinas será determinado por cada caso que estará sendo apresentado. O capítulo 4 apresenta os resultados da aplicação do modelo para dois clusters. O primeiro cluster foi projetado com base nos números do cluster do Google. O segundo cluster apresentado é um cluster com menos máquinas, mas cada uma com capacidade individual maior. O capítulo 5 apresenta as conclusões e indica direções para trabalhos futuros.

CAPÍTULO 2
CONCEITOS E TRABALHOS RELACIONADOS

Neste capítulo são demonstrados alguns conceitos importantes para o entendimento deste trabalho. Noções sobre teoria de filas, definições de desempenho e informações sobre servidores Web serão apresentados nas seções a seguir.

2.1 Teoria de Filas

Filas são comuns no nosso dia-a-dia. Pessoas que ligam para tele-atendimentos ou vão a bancos ou cinema, todos estão sujeito a enfrentar filas para serem atendidos.

A teoria das filas usa um conceito matemático de filas de espera. A formação destas é um acontecimento comum que ocorre sempre que a demanda atual por um determinado serviço excede a capacidade atual de atender este serviço.

Para especificar o sistema de filas, foi criado um padrão que representa todas as características que podem ser especificadas neste sistema. Este padrão é chamado de notação de Kendall [13] e é composto por seis parâmetros. A sua representação tem a seguinte forma:

A / S / m / B / K / SD

onde cada parâmetro pode ser especificado como a seguir.

O parâmetro A representa o processo de chegada, que especifica qual a distribuição que será respeitada para saber o comportamento dos processos de chegada. O processo de chegada mais utilizado é chamado de chegadas de Poisson, que significa que o intervalo entre as chegadas é independente e identicamente distribuído (IID), respeitando uma distribuição exponencial.

O tempo de serviço é o tempo que o sistema gasta atendendo uma requisição. É comum que a distribuição utilizada neste parâmetro também seja IID. A distribuição do tempo de serviço é representada pela variável S.

O número de servidores é representado pela variável m, que informa qual o total de servidores que o sistema de filas dispõe para atender as requisições dos clientes.

A variável B representa a capacidade do sistema, que é a quantidade máxima de requisições que o sistema pode suportar.

O tamanho da população, definido pela variável K, é a quantidade máxima de clientes que podem potencialmente enviar requisições ao sistema. Na maioria dos sistemas reais, a população é limitada. Porém, para facilitar os cálculos, a teoria de filas pressupõe na maioria das vezes que a população é infinita.

A disciplina de serviço SD é a forma como as requisições são atendidas. A disciplina mais utilizada é a *First Come, First Served* (FCFS), onde as requisições são atendidas na ordem de chegada.

Quando o sistema utilizado tem seus parâmetros B e K definidos como infinitos e a disciplina de serviço como FCFS, somente os três primeiros parâmetros da notação de Kendall são obrigatórios. Neste caso, para especificar um sistema com três servidores e processos de chegada e disciplina de serviço respeitando uma distribuição exponencial, a notação a ser utilizada é M/M/3. Os tipos de fila mais comuns são os seguintes:

- M/M/1: Sistema que possui apenas um servidor e tempos entre chegadas e de serviço seguindo a distribuição exponencial.
- M/M/m: Sistema que possui m servidores e distribuição exponencial para tempos entre chegada e de serviço.
- M/M/m/B: Igual ao sistema anterior, com a diferença de que somente um número B de requisições pode estar presente no sistema ao mesmo tempo.
- M/G/1: É o sistema que será utilizado neste trabalho. Possui apenas um servidor, os processos de chegada respeitam a distribuição exponencial e o tempo de serviço segue uma distribuição genérica.

A maioria dos modelos de filas se baseia em processos aleatórios, conhecido como processos de Markov [13], e dependem dos tipos de distribuição de probabilidade para as taxas de chegada e de serviço. As distribuições mais comuns são mostradas na Tabela I.

TABELA I
DISTRIBUIÇÕES MAIS UTILIZADAS NOS SISTEMAS DE FILAS

Símbolo	Distribuição
M	Exponencial
E_k	Erlang com parâmetro k
H_k	Hiperexponencial com parâmetro k
D	Determinística
G	Genérica

Os processos de Markov são definidos por processos que não dependem do que aconteceu com o sistema no passado. Apenas o que acontece no presente é significativo para o sistema. Sendo assim, processos futuros não necessitam saber dos estados passados para serem executados e nem são afetados por tais acontecimentos.

A teoria de filas também utiliza algumas variáveis para facilitar a recuperação de informações e que permitem a análise das filas que estão sendo estudadas. Estas variáveis são encontradas na Tabela II.

TABELA II
VARIÁVEIS DE UM SISTEMA DE FILAS

Variável	Significado
τ	Tempo entre duas chegadas sucessivas
λ	Taxa de chegada média
s	Tempo de serviço por requisição
μ	Tempo médio de serviço por servidor
n	Número de requisições no sistema
n_q	Número de requisições na fila
n_s	Número de requisições em serviço
r	Tempo de resposta
w	Tempo de espera na fila
U	Utilização do sistema

A Tabela III resume os principais parâmetros e fórmulas utilizados no sistema de fila M/G/1. Estas informações podem ser encontradas em [13, 15]. Algumas destas fórmulas foram utilizadas como base para a criação do modelo que é apresentado neste trabalho.

TABELA III
FÓRMULAS PARA A FILA M/G/1

1 - Parâmetros: λ = Taxa de chegada de requisições por unidade de tempo S = Tempo médio de serviço por requisição C = Coeficiente de variação do tempo de serviço U = Utilização do sistema
2 – Intensidade de tráfego: $U = \lambda \times S$
3 – O sistema está estável quando U < 1.
4 – Probabilidade de não haver nenhuma requisição no sistema: $U_0 = 1 - U$.
5 – Número médio de requisições no sistema: $n_r = U + U^2(1+C^2) / [2(1 - U)]$
6 – Variância do número de requisições no sistema: $Var[n_r] = n_r + \lambda^2 Var[S] + \frac{\lambda^3 S^3}{3(1-U)} + \frac{\lambda^4 (S^2)^2}{4(1-U)^2}$
7 – Número médio de requisições na fila: $n_q = U^2(1+C^2) / [2(1 - U)]$
8 – Variância do número de requisições na fila: $Var[n_q] = Var[n_r] - U + U^2$
9 – Tempo médio de resposta: $T = n_r / \lambda + U \times S(1+C^2) / [2(1 - U)]$

Um dos fundamentos principais da teoria de filas é conhecido como Lei de Little [13, 19]. A Lei de Little, que foi provada por Little em 1961, é baseada na idéia de uma caixa preta, onde o número de requisições que entram no sistema é o mesmo que são completados pelo sistema. Nenhuma requisição é criada ou perdida

dentro desta caixa preta. Esta lei relaciona o número médio de requisições em qualquer sistema com o tempo médio em que uma requisição permanece no sistema de acordo com a equação abaixo:

Número médio de requisições no sistema =
taxa de chegada x tempo médio de resposta

2.2 Métricas de desempenho

As métricas são definições de valores quantitativos que visam determinar o estado de funcionamento de uma determinada estrutura. Podem também ser utilizadas para comparar dois ou mais sistemas.

Para cada métrica, existe uma faixa de valores que é considerada a faixa de normalidade esperada para uma métrica fornecida. Para que comparações possam ser feitas e que seja possível decidir entre um sistema ou outro, é necessário escolher qual será a métrica-base. Sendo assim, essas métricas de desempenho irão auxiliar na decisão por um ou outro sistema baseado em quesitos pré-determinados. As medidas mais importantes usadas para avaliar os sistemas Web são tempo de resposta, taxa de processamento, disponibilidade e custo.

O tempo de resposta pode ser definido de diversos modos. Basicamente, esta métrica é o tempo que é gasto desde a requisição feita pelo usuário até a recepção da resposta fornecida pelo sistema. Para entender melhor como o tempo de resposta funciona, vamos detalhar o atendimento de uma requisição. No instante t_0, o cliente acabou de receber uma resposta do servidor. Entre os instantes t_0 e t_1, o usuário está decidindo o que fazer em seguida e se prepara para enviar o próximo pedido. O intervalo entre t_0 e t_1 é chamado de tempo de pensar. No instante t_1, o cliente envia um novo pedido ao servidor. A resposta ao servidor começa a chegar no cliente no instante t_2 e termina no instante t_3. Os dois intervalos (t_2-t_1 e t_3-t_1) normalmente são chamados de tempo de resposta. Para distinguir entre os dois, usaremos o termo tempo de reação para indicar o intervalo t_2-t_1. Considere o caso de um usuário navegando pela Web. No momento em que o usuário clica em um link no navegador, um pedido é enviado ao servidor. Esse pedido pode fazer com que vários arquivos, incluindo textos e imagens, sejam

recuperados do servidor e exibidos pelo navegador. Assim que o primeiro documento começa a ser exibido, o tempo de reação termina. Quando todo o texto e os arquivos de imagens são completamente carregados, o intervalo do tempo de resposta termina. Informações detalhadas sobre tempo de resposta de ponta a ponta podem ser encontradas na referência [14]. O tempo de espera é a medida de tempo em que o sistema fica aguardando por requisições.

O *throughput* é a métrica definida como o número de requisições executadas por unidade de tempo. A unidade utilizada depende do tipo da requisição que está sendo enviada ao servidor. Para servidores Web, uma medida utilizada pode ser a quantidade de requisições HTTP servidas por segundo, como exemplo.

A métrica custo normalmente está associada a alguma medida de desempenho, como tempo de resposta ou *throughput*. O valor do custo deve incluir tanto os valores gastos com hardware quanto com software.

Os custos são subdivididos em duas categorias: iniciais e operacionais. Os custos iniciais são aqueles incorridos na instalação do sistema, enquanto que os custos operacionais são as despesas para a conservação e manutenção do sistema e oferecer melhorias de hardware e software, a fim permitir a adequação do sistema às novas cargas empregadas e correções de vulnerabilidades. Os custos iniciais aplicam-se a despesas de instalação, hardware, software, infra-estrutura e desenvolvimento de conteúdo. Os custos operacionais estão relacionados à manutenção e a ampliação do hardware e software, custo de pessoal, treinamento, despesas com consultoria, entre outros serviços de terceiros. Normalmente para a comparação de sistemas é utilizada a notação de quantas requisições por segundo uma quantidade de dólares pode comprar. Por exemplo, um sistema é capaz de executar 1.500 requisições por segundo e seu valor de mercado é de US$ 6.000,00. Então podemos dizer que para este sistema são necessários US$ 4 para comprar uma requisição por segundo.

Uma metodologia de planejamento de capacidade exige a identificação das principais origens do custo, além da determinação de como os custos variam com o tamanho e a arquitetura do sistema. Ao avaliar os custos incorridos pelos novos serviços Web, é preciso considerar que, em alguns casos, a introdução de novos serviços, como a adição de um servidor de FTP ou SMTP por exemplo, pode

aumentar o tráfego na rede já existente [22] , gerando uma maior demanda de acesso ao servidor.

Quando o modelo de desempenho estiver montado e solucionado, e um modelo de custo tiver sido desenvolvido, diversas análises podem ser realizadas com relação às escolhas de custo-desempenho. Os modelos de desempenho e os modelos de custo podem ser usados para avaliar diversos cenários e configurações. Para cada cenário, podemos prever qual será o desempenho de cada componente básico da carga de trabalho global e quais são os custos para aquele cenário.

A utilização do sistema mede a quantidade de recursos que estão sendo empregados para a execução de uma determinada tarefa. Esta utilização pode ser a do sistema como um todo ou somente de uma parte dele, como CPU, espaço em disco ou memória, por exemplo.

A confiabilidade pode ser definida como a probabilidade de que um sistema fique operante por um determinado período de tempo T e respondendo às requisições da forma esperada. Quanto maior o tempo em que o sistema está *on-line* e respondendo a requisições de acordo com o que é solicitado, maior a sua confiabilidade. Se o sistema consegue responder às requisições, mas de forma incorreta, ele não é considerado confiável. A disponibilidade é a fração de tempo em que um sistema fica operacional, ou seja em execução. Quando a variável T assume um valor grande, tendendo ao infinito, a confiabilidade torna-se igual à disponibilidade [27].

Existem diversos motivos para um sistema falhar. Estes motivos podem ser classificados em diferentes tipos de falhas compostas por três tipos: duração, efeito e escopo.

As falhas de duração ocorrem quando o sistema pára e nenhum de seus recursos pode ser acessado. Estas falhas podem ser subdivididas em falhas permanentes, recuperáveis e transientes. As falhas permanentes ocorrem quando o sistema não pode ser recuperado de seu ponto de falha, onde mesmo a intervenção humana não consegue corrigir o problema. Quando o sistema volta de uma falha após alguns reparos, o sistema passou por uma falha recuperável. Se nenhuma ação foi tomada e o sistema voltou a funcionar normalmente, pode-se dizer que a falha a qual o sistema passou foi do tipo transiente.

Falhas de efeito não fazem com que o sistema pare completamente, porém suas especificações não são cumpridas de acordo com o esperado. Este tipo

de falha pode ser classificado em dois tipos: funcionais e de desempenho. A falha é funcional quando o sistema não opera de acordo com suas especificações, ou seja, uma ou mais de suas funcionalidades deixa de operar. Um exemplo é um *site* de compras on-line onde a opção de detalhar o produto não esteja acessível. As falhas de desempenho ocorrem quando o sistema não é capaz de responder em um tempo hábil. Serviços onde o tempo de resposta é imprescindível, como em *sites* de *streaming*, podem ter uma falha de desempenho quando o vídeo não é exibido de forma contínua.

A última classe de tipos de falha é conhecida por falhas de escopo, podendo ser divididas em dois tipos: falhas parciais ou totais. Uma falha é parcial quando o sistema pode ser acessado parcialmente, onde seções do sistema continuam funcionando normalmente, sem interrupções. Falhas totais causam a parada por completo do sistema, onde todas as funcionalidades ficam inacessíveis.

Para medir a disponibilidade nos sistemas computacionais, é utilizada a teoria dos “9”s, onde quanto mais dígitos 9 a definição possuir, maior a disponibilidade do sistema. A Tabela abaixo mostra as classes de disponibilidade e qual o tempo máximo que o sistema pode ficar indisponível dentro do período de um ano.

TABELA VI
CLASSES DE SISTEMAS DE ACORDO COM A DISPONIBILIDADE

Classe	Disponibilidade	Indisponível (min/ano)	Tipo de Sistema
1	90%	52.560	Não controlado
2	99%	5.256	Controlado
3	99,9%	526	Bem Controlado
4	99,99%	52,6	Tolerante a falhas
5	99,999%	5,3	Altamente disponível
6	99,9999%	0,53	Muito altamente disponível
7	99,99999%	0,0053	Ultradisponível

A confiabilidade do sistema pode ser calculada para dois tipos distintos de sistemas: os sistemas em série e os sistemas em paralelo. Um sistema é considerado em série quando todos os seus componentes são indispensáveis para o seu total funcionamento. Um sistema é paralelo quando os componentes trabalham integrados, mas são independentes entre si.

Para calcular a probabilidade de um sistema serial estar em funcionamento, todos os seus n componentes devem estar em funcionamento. Assim, a probabilidade neste caso é o produto de todas as probabilidades de todos os componentes. Sendo R_s a confiabilidade do sistema serial e r_n a confiabilidade do componente n, temos que:

$$R_S = \prod_{i=1}^{n} r_i$$

Se forem considerados todos os componentes iguais, então podemos dizer que $R_s = r^n$. Para este caso, quanto mais componentes tiver o sistema, maior a probabilidade de se ter uma falha. A Figura 1 mostra um sistema em série.

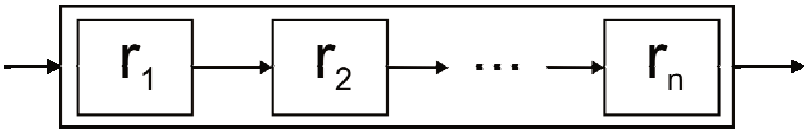

Figura 1 – Sistema em série

Para sistemas em paralelo se um dos componentes falha, os demais não são afetados. Neste caso, a confiabilidade do sistema pode ser calculada através da probabilidade de que exista algum dos componentes em funcionamento. Para que o sistema pare, todos os n componentes precisam estar em falha. A probabilidade para que o componente n esteja em falha é de $(1 - r_n)$. Assim, considerando a interdependência das falhas entre componentes, temos que $R_p = 1 -$ [Probabilidade dos componentes parados]. Sendo assim podemos dizer que $R_p = 1 - [(1 - r_1) \times (1 - r_2) \ldots \times (1 - r_n)]$, concluindo então que:

$$R_P = 1 - \prod_{i=1}^{n} (1 - r_i)$$

Considerando-se que todos os componentes são iguais, podemos dizer que $R_p = 1 - (1 - r)^n$. Para os sistemas paralelos, quanto mais componentes o sistema possui, maior a sua confiabilidade. Um modelo de sistema em paralelo pode ser visualizado na Figura 2 abaixo.

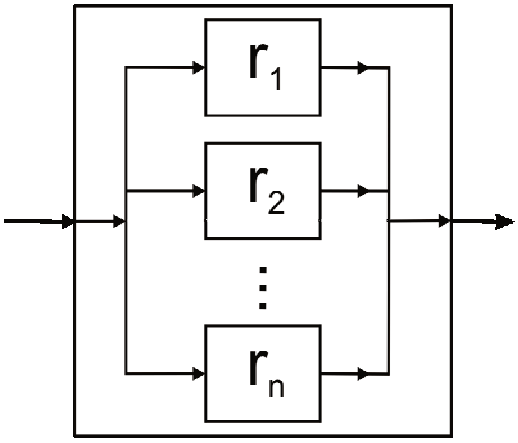

Figura 2 – Sistema em paralelo

Considerando que existem diferentes sistemas computacionais e diferentes cargas de trabalho possíveis, é necessário escolher a métrica a ser utilizada de acordo com os objetivos a serem analisados, pois os resultados obtidos para comparação podem ser diferentes de acordo com a escolha que for feita. Também é necessário saber como comparar o valor obtido na análise. Desta forma, as métricas de desempenho podem ser classificadas em três tipos: maior valor é melhor, menor valor é melhor e valor nominal é melhor.

No primeiro caso, quanto maior o valor encontrado na análise da métrica, melhor é o desempenho do sistema que está sendo verificado. Um exemplo de métrica que segue este padrão é a disponibilidade, pois quanto mais tempo o sistema fica disponível, melhor ele é. O segundo caso ocorre quando a métrica assume menores valores e o desempenho é melhorado. Um exemplo de métrica para este caso é o tempo de resposta, onde quanto menor o tempo em que o servidor responde, melhor é o sistema. No terceiro caso, nem os menores nem os maiores valores são os mais importantes. Um valor mediano pode representar um melhor desempenho. Um exemplo de métrica é a utilização, onde valores entre 50 a 75% podem ser considerados melhores.

2.3 Servidores Web

O servidor Web, também conhecido por servidor HTTP ou *daemon* HTTP, é o software responsável pela implementação do protocolo HTTP (*Hyper Text Transfer Protocol*) que responderá as requisições dos clientes controlando o fluxo de dados recebidos e emitidos em um computador conectado a uma intranet ou à

Internet. O servidor Web pode ser considerado ainda como uma combinação de uma plataforma de hardware, sistema operacional, software servidor e conteúdo. A Figura 3 mostra a estrutura de um servidor Web. Basicamente, o servidor Web tem a função de aceitar requisições HTTP e retornar respostas, que podem ser páginas estáticas, conteúdo dinâmico ou mensagens de erro.

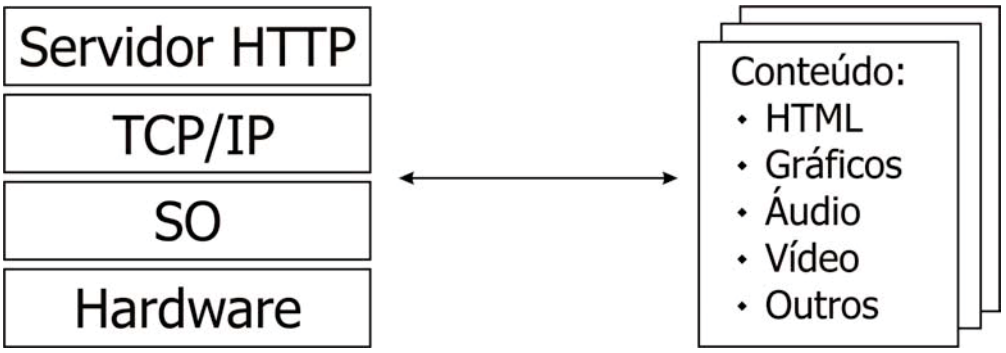

Figura 3 – Estrutura básica de um servidor Web

Para que um servidor Web seja acessado, normalmente é utilizado um software na máquina do cliente conhecido como *Web browser* ou navegador Web. Este navegador é o software que age como cliente dos servidores Web gerando requisições HTTP e é responsável pela implementação de todas as interações que serão feitas com o servidor.

Uma transação típica utilizando o protocolo HTTP é realizada de acordo com as seguintes etapas: o servidor aguarda por requisições em uma porta de comunicação específica, que é definida por padrão como a porta 80, até que uma conexão TCP seja estabelecida. Após a abertura da conexão TCP, o cliente realiza uma ou mais requisições e o servidor Web responde ao cliente o resultado da requisição. Esta resposta inclui um código de *status* que informa se a solicitação foi bem sucedida. Caso contrário, o motivo da falha de serviço é enviada ao cliente. Após o envio da resposta e finalização da conexão TCP, o servidor repete o ciclo e fica aguardando por novas requisições [30].

A maioria dos documentos Web é escrita em uma linguagem de marcação conhecida por HTML. O *Hyper Text Markup Language* permite que se crie de forma estruturada documentos com título e links para outros documentos incluindo imagens e outros objetos multimídia criando uma integração de vídeo, aplicações de softwares e objetos multimídia na Web. Quando um navegador analisa

os dados HTML recebidos de um servidor, ele reconhece os hyperlinks associados às imagens e automaticamente solicita os arquivos ao servidor. Embora o usuário veja apenas um documento, o servidor vê uma série de pedidos separados para esse documento. Em termos de desempenho, é importante entender que um único clique por um usuário pode gerar uma série de pedidos ao servidor.

Muitos servidores Web, além de páginas HTML estáticas, podem gerar dinamicamente o conteúdo de acordo com as requisições dos clientes. Tecnologias Web como Java, ASP, ActiveX ou DHTML são algumas das formas de gerar conteúdo dinâmico na Web. Essas tecnologias também permitem a inclusão de efeitos especiais nos documentos sem a necessidade de contar com programas extras no servidor. O grande problema do conteúdo dinâmico para os servidores é a geração de carga extra para processar os códigos gerados pelos criadores dos sites. É possível ainda ter páginas dinâmicas utilizando scripts que são executados no lado do cliente. Um exemplo deste tipo de aplicação é o Javascript.

Para melhorar o desempenho e atender mais clientes por vez, os servidores podem utilizar três técnicas: geram uma cópia do processo HTTP para cada pedido, realizam o atendimento com *multithreading* ou espelham os pedidos em um *pool* de processos.

O aumento no tráfego na Web fez com que os servidores ficassem mais sobrecarregados, incentivando a ampliação dos esforços para que o desempenho dos servidores fosse otimizado. Além do tráfego, o tempo de resposta de um servidor Web pode tornar-se lento por diversos motivos. Entre eles, podemos citar: a configuração do hardware onde o servidor Web está sendo executado, a configuração do software do servidor Web, a largura de banda disponível, a estrutura e a configuração das aplicações Web, as características dos computadores dos usuários e as suas respectivas conexões, além do aumento na carga submetida aos servidores.

Nem todos os fatores citados acima podem ser controlados pelo administrador do servidor. É possível que o administrador tenha pouca influência sobre a estrutura das suas aplicações para a Web, mesmo quando estas são desenvolvidas pela própria equipe do administrador. Porém, há ainda muito que está ao seu alcance. O próprio servidor pode ser o ponto de partida para a melhoria do desempenho.

O desempenho do servidor depende principalmente do comportamento de seus componentes básicos: a plataforma de hardware e seu sistema operacional. O desempenho de um servidor é função principalmente da velocidade de seus processadores, da quantidade de memória principal, largura de banda da rede, da capacidade de armazenamento e da rapidez com que as informações são recuperadas no disco e enviadas ao cliente.

Os servidores Web podem executar em diversos sistemas operacionais, como Unix, Linux, Windows NT ou 2000, por exemplo. É importante ressaltar que com a escolha do sistema operacional, estão também sendo decididos quesitos como confiabilidade e escalabilidade, assim como a implementação do protocolo TCP/IP, que é muito importante para o desempenho do HTTP.

O conteúdo armazenado no servidor também pode ter grande influência no desempenho do servidor e gerar gargalos de acordo com os documentos ali armazenados e de quais arquivos estão sendo requisitados. Um servidor que tenha muitas requisições de arquivos grandes pode ter seu disco ou a largura de banda da rede como gargalo, enquanto que servidores que sirvam conteúdo dinâmico precisam ter processadores mais velozes para suprir as necessidades de processamento.

2.3.1 Pilha de Protocolos

A comunicação entre dois computadores é controlada por um conjunto de regras chamado protocolo. Se duas entidades (X e Y) precisam se comunicar por uma rede, X precisa endereçar corretamente o destino para que as mensagens consigam chegar a Y, que é o objetivo esperado. As funções de roteamento, endereçamento, controle de seqüência dos pacotes, controle de fluxo e detecção e recuperação de erros são funções do protocolo.

O projeto de um protocolo de comunicação é uma tarefa bastante complexa. Por isso, normalmente os projetistas usam uma estrutura em camadas para facilitar a implementação do protocolo. A International Organization for Standardization (ISO) criou um modelo em sete camadas, conhecido como modelo OSI (*Open Systems Interconnection Reference Model*) [31] para facilitar implementações e melhorar a interligação entre diferentes serviços. O objetivo deste modelo é fornecer uma base comum que permita o desenvolvimento coordenado de

padrões para interconexão de sistemas. Este modelo proposto não interfere nos projetos de arquitetura de rede, mas especifica qual a tarefa que cada uma das camadas deve realizar. A Figura 4 mostra o esquema das camadas do modelo OSI.

A camada física estabelece as características mecânicas, elétricas, funcionais e os procedimentos para ativar, manter e desativar conexões físicas para a transmissão de bits entre entidades do nível de enlace.

A camada de enlace é capaz de detectar e opcionalmente corrigir os erros gerados pela camada física, tornando um meio não confiável em um meio confiável para uso da camada de rede. A técnica comumente utilizada é a redundância e a divisão dos pacotes em quadros.

A camada de rede fornece subsídio à camada de transporte para que ela seja independente quanto a considerações de chaveamento e roteamento associado ao estabelecimento e operação de uma conexão de rede. Esta camada não garante que os pacotes serão entregues nem a ordem na qual o destinatário irá recebê-los.

A camada de transporte possui a característica de conseguir manter uma conexão fim-a-fim. Controla também o fluxo de informações para que o buffer utilizado não seja extrapolado. Possui a capacidade de reordenar os pacotes que a camada de rede não foi capaz de organizar.

A camada de sessão fornece mecanismos que permitem estruturar os circuitos oferecidos pela camada de transporte. Os serviços mais comuns desta camada são: gerenciamento de *token*, controle de diálogo e gerenciamento das atividades.

A camada de apresentação realiza as transformações necessárias nos dados antes de enviá-los ao nível de sessão. Os serviços de compressão de texto, criptografia, conversão de padrões são algumas das tarefas realizadas nesta camada.

A camada de aplicação fornece aos processos de aplicação os meios para que estes utilizem as demais camadas do modelo OSI. São fornecidos também recursos que possibilitem a utilização de aplicações distribuídas.

Figura 4 – Camadas do modelo OSI

Uma especificação de protocolo consiste em dois elementos: a sintaxe e a semântica. A sintaxe do protocolo especifica todas as mensagens trocadas entre as entidades, seus formatos e o significado de cada campo da mensagem. A semântica de um protocolo especifica as ações a serem tomadas por cada entidade quando ocorrem eventos específicos, como chegada de mensagens ou *timeout*.

Os protocolos mais importantes que trafegam na Internet são o *Internet Protocol* (IP) e o *Transmission Control Protocol* (TCP). Estes dois protocolos são conhecidos conjuntamente como TCP/IP e serão detalhados a seguir.

O protocolo IP foi projetado para permitir a interconexão de redes de computadores que utilizam a tecnologia de comutação de pacotes. Ele especifica os formatos dos pacotes que trafegam pela Internet e os mecanismos utilizados para encaminhar estes pacotes entre uma coleção de redes e roteadores entre a origem e o destino.

Cada máquina que utiliza este protocolo possui um endereço exclusivo, chamado de endereço IP. Este endereço, como é mais utilizado atualmente, possui 32 bits e é representado por um conjunto de quatro números separados por pontos, como em 192.168.0.1. Esta versão do IP é conhecida como IPv4 (IP versão 4). Porém como o número de *hosts* utilizados atualmente cresce de forma exponencial, a quantidade de endereços IP disponíveis está se tornando insuficiente. Para isso, foi desenvolvido o IP versão 6 (IPv6), que amplia o tamanho do endereço de 32 para 128 bits, aumentando assim o número de endereços disponíveis.

Cada um dos quatro números de identificação do endereço Ipv4 representa 8 bits e pode ter seu valor decimal definido entre zero e 255. Estes 32 bits são divididos em duas partes. A primeira parte chamada de prefixo, indica a rede na qual o *host* está inserido. E a segunda parte, o sufixo, indica um *host* dentro da rede. O número de bits alocados ao prefixo determina a quantidade de números de rede exclusivos e o número de bits do sufixo determina o número de *hosts* por rede.

Os endereços IP podem ser usados para referir tanto a redes quanto a cada *host* individualmente. O endereço de rede tem o campo identificador do *host* com todos os bits iguais a zero. Podemos nos referir também a todos os hosts da rede através de um endereço de difusão, onde o campo identificador dos hosts deve ter todos os bits iguais a 1. Além disso, os endereços IP foram divididos em classes de utilização. A definição de classes se deve ao fato de que o tamanho das redes pode variar muito, indo desde redes locais de pequeno porte a redes de milhares de *hosts*.

Os endereços de classe A possuem o bit mais significativo igual a zero e são utilizados em redes de grande porte. Os endereços de rede variam de 1 a 126, onde cada rede pode endereçar cerca de 16 milhões de *hosts*. A classe B utiliza dois octetos para o número de rede e dois para os endereços de host. Os endereços desta classe variam de 128.1 a 191.255, podendo cada rede possuir cerca de 65 mil hosts. Os números 0 e 255 no segundo octeto e 127 no primeiro octeto não são utilizados nas configurações de rede tendo função diferenciada. A classe C utiliza os três primeiros octetos para representar a rede e apenas um para representar os hosts. Os endereços podem variar de 192.1.1 a 223.254.254 e cada rede pode possuir 254 hosts. Os endereços acima de 223 foram reservados para uso futuro.

A unidade de dados transportada pelo IP é o datagrama. O IP não oferece uma garantia de que os pacotes serão entregues no destino. Sendo assim, pacotes podem ser perdidos ou até mesmo serem entregues fora de ordem, pois cada datagrama é tratado como uma unidade independente que não possui relações com qualquer outro datagrama. Para que estes problemas sejam solucionados, o IP trabalha em conjunto com o protocolo TCP para que este cuide da recuperação de erros e a organização de seqüência de ponta a ponta. Uma função importante do IP é o roteamento de datagramas da origem ao destino. Para isso, um cabeçalho de 20 bytes de extensão é utilizado no protocolo IP, onde quatro bytes são usados para o endereço IP do *host* de origem e quatro bytes para o endereço IP de destino.

O protocolo TCP é um protocolo que oferece um serviço orientado a conexão controlando o fluxo até o destino que garante que os dados serão entregues na ordem em que foram transmitidos sem falta de informações. Ele permite que as duas pontas da conexão troquem dados simultaneamente em modo full-duplex e que um fluxo contínuo de bytes possa ser enviado em uma mesma conexão. Para que os erros encontrados durante a transmissão sejam detectados, o protocolo TCP se utiliza de confirmações (ACKs), *timeouts* e retransmissões de pacotes.

O TCP interage de um lado com processos das aplicações e do outro com o protocolo da camada inter-rede da arquitetura Internet. A interface entre os processos de aplicação e o TCP consiste em um conjunto de chamadas semelhantes às que os sistemas operacionais fornecem aos processos de aplicação para manipulação de arquivos. Sendo assim, existem chamadas para abrir ou encerrar uma conexão e para enviar e receber dados em conexões previamente estabelecidas.

O TCP estabelece uma conexão entre os dois *hosts* para que ocorra a efetiva troca de dados. Um mecanismo nomeado de *handshake* triplo foi implementado para garantir a confiabilidade da conexão. Na Figura 5 é mostrado como uma conexão TCP se estabelece.

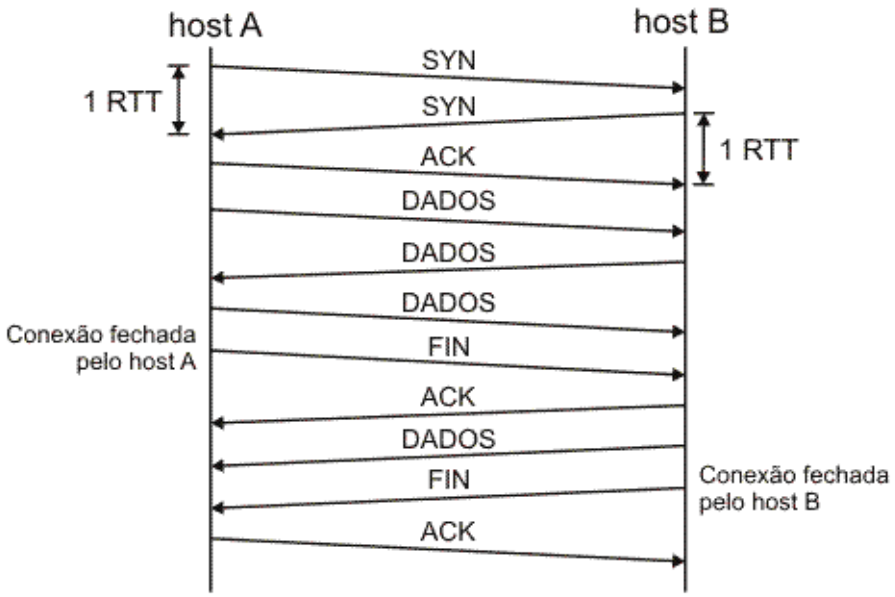

Figura 5 – Estabelecimento e finalização de uma conexão TCP

Para exemplificar como ocorrem as conexões TCP, mostraremos o processo de comunicação entre os *hosts* A e B. Primeiramente o *host* A envia uma mensagem de sincronização (SYN) a B informando que deseja trocar informações com ele. O *host* B envia outro SYN para indicar que está ciente do pedido de A. Esta troca de segmentos SYN ocupa um tempo de ida e volta (RTT). A conexão no *host* B não é considerada completa até que um segmento ACK seja recebido do *host* A. Durante o intervalo de tempo que o *host* B ficou aguardando a confirmação de A, a conexão fica em uma fila de conexões incompletas. Uma conexão fica nesta fila por um RTT. Então, quando o segmento ACK é recebido no *host* B, o processo de conexão é completado. É este processo que é conhecido como *handshake* triplo. Uma observação neste modelo é que se um *host* recebe muitas conexões de vários servidores remotos quase que ao mesmo tempo, a fila de conexões incompletas pode ficar cheia, impedindo novas conexões. Esta característica pode ser aproveitada por *hackers* para realizar ataques de negação de serviço (DoS - *Denial of Service*), onde o servidor fica tratando requisições falsas e não consegue processar as requisições dos legítimos clientes.

Para encerrar a conexão, o servidor deve enviar um segmento FIN. Na Figura acima, o *host* A fecha a conexão com B enviando um segmento FIN, que é confirmado com um segmento ACK. Quando o *host* B deseja encerrar a conexão com A, ele deve enviar o segmento FIN e aguardar o envio do segmento ACK pelo *host* A, confirmando o encerramento da conexão. Assim, três segmentos são necessários para iniciar uma conexão e quatro para termina-la nas duas direções.

HTTP é o protocolo em nível de aplicação usado na comunicação entre clientes e servidores Web. O protocolo HTTP define uma única interação de requisição e resposta, que pode ser vista como uma transação Web. Cada interação consiste em uma requisição enviada do cliente ao servidor, seguida por uma resposta enviada de volta pelo servidor ao cliente. Uma requisição HTTP inclui várias partes: o método que especifica a ação (GET, HEAD, PUT, POST), a URL (*Uniform Resource Locator*) que identifica o nome da informação solicitada, além de outras informações, como o tipo de documento que o cliente deseja aceitar e a autenticação.

Quando um servidor recebe uma requisição, ele a analisa e efetua a ação especificada pelo método. O servidor responde ao cliente utilizando uma linha de status, indicando o sucesso ou falha da requisição, meta-informações

descrevendo o tipo do objeto retornado e a informação solicitada, em um arquivo ou uma saída gerada por uma aplicação no servidor.

O HTTP é um protocolo implementado na camada de aplicação e não possui estados, pois não utiliza o conceito de sessão em suas transações. No protocolo HTTP original, na versão 1.0, cada requisição é independente da requisição anterior ou da seguinte. A conversação é restrita à transferência de um documento ou imagem. A vantagem deste método é que o servidor Web não precisa registrar quem são os clientes ou quais requisições foram atendidas anteriormente. A desvantagem é que com a utilização de muitos textos e imagens pequenos, são necessárias várias conexões para que uma única página Web seja exibida, aumentando o número de pacotes de controle TCP. Com a versão 1.1 do protocolo isto não ocorre, pois ele utiliza o conceito de conexão persistente, onde em uma mesma conexão vários arquivos diferentes podem ser enviados ao cliente. A versão 1.0 do HTTP está documentada na RFC 1945 [7] e foi padrão nos *browsers* e servidores até 1997. A versão 1.1 está especificada na RFC 2068 [37]. O HTTP 1.1 é compatível com 1.0, ou seja, clientes e servidores podem trocar informações mesmo implementando versões distintas do HTTP. Na Figura 6 é possível ver a relação entre os protocolos HTTP, TCP e IP.

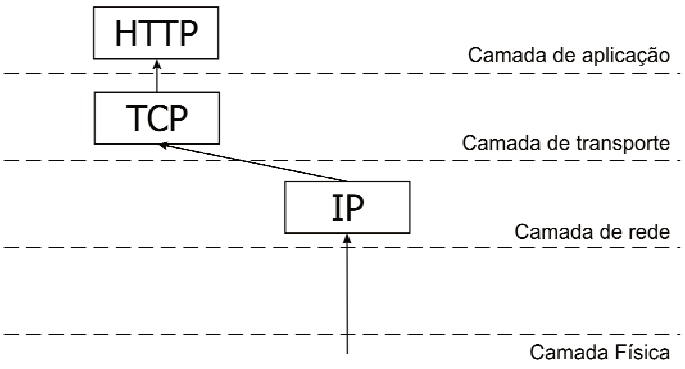

Figura 6 – Relação entre os protocolos

Uma outra característica do protocolo HTTP é a sua assimetria. O fluxo de dados no sentido do servidor para o cliente é bem maior do que o fluxo no sentido inverso. Assim, em relação à quantidade de dados, o gargalo tende a se localizar mais na saída do que na entrada do servidor. Podemos dizer também que o protocolo HTTP é baseado em texto. Todas as trocas de informações são

estabelecidas através de mensagens em formato texto que o servidor e o navegador trocam entre si.

2.3.2 Melhorias nos servidores Web

Existem muitas propostas de melhorias dos servidores sendo estudadas, incluindo-se diversas formas de medir o desempenho para descobrir o comportamento apresentado quando uma determinada carga é empregada. Alguns autores fazem comparações utilizando servidores e sistemas operacionais diferentes e observando suas particularidades. Um estudo completo utilizando esta metodologia pode ser observado em [18], que descreve experimentos comparando tempo de resposta e *throughput* de servidores Web em alguns sistemas Unix.

Os resultados obtidos neste trabalho mostram que detalhes de plataforma e o software do servidor não exercem grande influência no desempenho, porém quando grandes arquivos precisam ser enviados pela rede, é justamente o tempo de transferência que domina o tempo global do processo. Foi verificado também que *scripts* podem ocasionar um *overhead* significativo no tempo de resposta. Por fim, os servidores *multithreading* e os que utilizam *pool* de processos têm um desempenho consideravelmente melhor do que os que criam um atendente (*fork*) para executar cada requisição.

Outros trabalhos propõem ferramentas no intuito de medir uma ou mais métricas de desempenho. Entre as ferramentas, encontra-se o Webmonitor [2], capaz de obter diversas medidas como, por exemplo, o *throughput*. Estas ferramentas e técnicas de medições são importantes para caracterizar o comportamento do sistema e observar sua evolução dentro de um contexto ou avaliar o que se pode esperar do sistema quando ele estiver trabalhando com uma determinada carga.

Os testes com o Webmonitor demonstraram que se o servidor Web estiver com sua carga elevada, ele passa cerca de 90% do tempo tratando requisições HTTP no núcleo do sistema. Além disso, conexões TCP que permaneçam abertas aumentam o tempo de serviço em um fator que pode variar entre 2 a 9. Foi observado também o aumento na utilização de três recursos do servidor: número de processos executando simultaneamente, aumento na utilização da CPU e de memória.

Uma outra ferramenta desenvolvida por pesquisadores é a Certes (*CliEnt Response Time Estimated by the Server*) [25]. Esta ferramenta estima, no servidor, o tempo de resposta que o cliente percebe em sua máquina quando faz uma requisição para um determinado servidor. Para a criação da ferramenta não foram necessárias adaptações no sistema operacional e servidor Web, pois a ferramenta utiliza informações disponíveis no sistema.

Os métodos empregados para estimar o tempo de resposta percebido pelo cliente seguem a seguinte expressão:

$$CLIENT\_RT = CONN\_FAIL + SYN\text{-}to\text{-}END + RTT$$

onde CONN_FAIL é o tempo gasto com tentativas frustradas de conexão, SYN-to-END é o tempo necessário para se estabilizar uma conexão TCP até que o primeiro pacote SYN seja recebido mais o recebimento da requisição HTTP do cliente. Por fim, o RTT é o tempo em que os pacotes percorrem o caminho de ida e volta entre o servidor e a máquina do cliente.

Para definir o valor de CONN_FAIL, dados estatísticos que estimam o tempo gasto com as tentativas de conexão são empregados. Os valores utilizados são obtidos através de informações armazenadas no servidor.

Os resultados demonstrados em [25] demonstram que a ferramenta alcança os objetivos que estavam sendo esperados, tendo como valores obtidos números aproximados aos percebidos pelos clientes. Foi observado também que a ferramenta insere um *overhead* insignificante no sistema onde está operando e pode ser utilizada mesmo quando em situações de sobrecarga do servidor.

A área de ferramentas possui um vasto campo de estudo a ser explorado. Porém, é necessário que sistemas operacionais e protocolos de rede sejam bem integrados aos servidores Web para que o desempenho global do sistema melhore consideravelmente e as ferramentas possam detectar os problemas a serem resolvidos.

Após a realização de testes preliminares, sejam estes feitos sob forma de observação ou com o uso de ferramentas, algumas providências podem ser tomadas no intuito de melhorar o desempenho global do sistema adaptando-o à carga real que será utilizada. No caso de servidores Web, estão sendo realizadas diversas pesquisas que demonstram resultados consideráveis na área. Alguns

exemplos podem ser consultados em [2, 10, 12]. As formas de se incrementar o desempenho podem ser classificadas nas áreas listadas abaixo e descritas a seguir:

- Melhorias de hardware;
- Adaptações no software do servidor;
- Uso de caches;
- Replicação de servidores.

Melhorias de Hardware são a forma mais simples de se incrementar o desempenho. Com a utilização de ferramentas ou simplesmente com a observação do sistema, é possível saber quais são as limitações do servidor e melhorá-las adicionando ou trocando o componente que limita que o desempenho seja o esperado. Assim, uma melhoria significativa pode ser observada logo após a inclusão do novo componente ao sistema.

Os servidores Web possuem extensos arquivos de configuração. Ajustando parâmetros nestes arquivos é possível que os servidores fiquem mais bem adaptados a carga a qual serão submetidos. Podem ser alterados também o código-fonte do servidor e do kernel do sistema operacional para adapta-lo às necessidades das requisições feitas ao servidor.

Caches distribuídos podem ser implementados através de caches no lado do cliente, servidores proxy ou servidores cache dedicados. Estes métodos ajudam a retirar a grande demanda do tráfego de um único servidor centralizado. Um problema da utilização de caches, no entanto, é a grande quantidade de conteúdo dinâmico existente hoje na Internet. Este tipo de conteúdo não tem como ser tratado eficientemente através de caches, o que dificulta a utilização desta técnica [12]. Entretanto, alguns estudos já estão sendo realizados nesta área.

A replicação de servidores consiste em distribuir a mesma informação do servidor em diversas máquinas trabalhando em conjunto, formando clusters [6]. Este conjunto de máquinas é normalmente interconectado por meio de uma rede de alta capacidade e é visualizado como uma única máquina pelos usuários. Em geral, cada nodo do cluster possui seu próprio disco e sistema operacional independente, onde cada um dos servidores fica encarregado de uma parcela do total de tarefas. Com essa divisão de tarefas, a carga individual que cada máquina tem que processar é reduzida, mantendo-se a carga global. Um maior detalhamento sobre clusters está presente na seção seguinte.

2.3.3 Clusters de Servidores Web

Por mais otimizado que seja um servidor Web, uma única máquina pode não conseguir preencher os requisitos de tolerância à falhas, escalabilidade e principalmente desempenho, demandados para estes sites. A arquitetura baseada em múltiplos servidores é uma alternativa efetiva e relativamente barata que tem sido adotada atualmente. Administradores dos sites estão sempre precisando expandir seus sistemas e a utilização de clusters é uma das opções disponíveis.

Normalmente, antes de se pensar em expandir o sistema com a adição de novos servidores, a primeira opção é a alteração ou adição de componentes no servidor já existente, como por exemplo a adição de mais memória ou a troca do processador. Esta solução pode ser nomeada como escalabilidade de hardware. Entretanto, esta solução pode não alterar significamente os recursos do servidor fazendo com que em um curto espaço de tempo os problemas de desempenho voltem a acontecer.

Outra solução que pode ser adotada é a modificação a nível de software, como demonstrada em alguns casos na seção anterior. Esta forma de escalonar é conhecida como escalabilidade de software.

Estes dois tipos de escalonamentos são classificados como escalabilidade vertical ou *scale-up*, onde apenas um servidor é responsável pelo atendimento as requisições dos clientes. Para este trabalho, o tipo de escalonamento dos servidores é a escalabidade horizontal ou *scale-out*, onde são utilizados múltiplos servidores replicados e a expansão do sistema acontece com a adição de novos servidores completos e não apenas de um ou outro componente.

Pode-se classificar técnicas de replicação de servidores em dois grupos: local e global. O primeiro se concentra na distribuição de carga sobre servidores distribuídos localmente, fisicamente próximos, sendo útil quando se atinge o limite de hardware de um servidor. Já o segundo se foca na utilização de servidores dispersos geograficamente e é adotado quando o problema de desempenho se concentra no limite da rede.

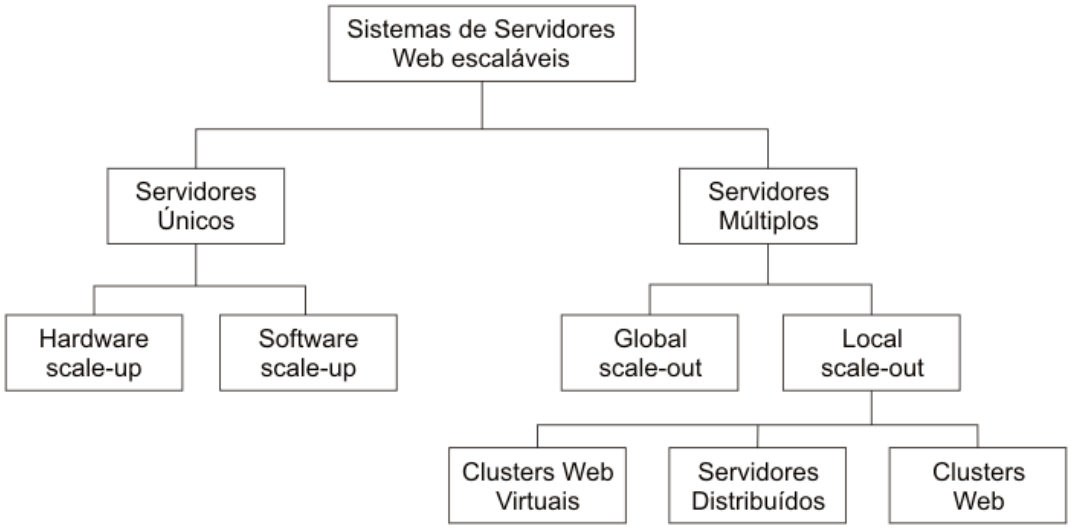

Figura 7 – Soluções de arquitetura para sistemas Web

A distribuição local de servidores, em um único ponto da Internet é útil para a situação onde um único *host* não está sendo capaz de atender a demanda de requisições, embora a rede suporte a carga.

Independente da implementação que for adotada para a construção do cluster, um sistema de servidores precisa parecer para os clientes como se fosse único, onde ele não precise se preocupar se está acessando o servidor A ou B. Sendo assim, o cliente não sabe como os servidores estão agrupados nem quantos servidores estão disponíveis.

A Web exibe extrema variabilidade nas características da carga de trabalho. Por exemplo, tem sido verificado na análise de alguns sites Web populares que existem poucos documentos cujo tamanho é inferior a 100 bytes [4]. A maior parte dos documentos tem entre 100 bytes e 100 kbytes, enquanto que alguns arquivos possuem tamanho maior que 100 kbytes. A distribuição dos tamanhos de arquivos na Web exibe um comportamento de cauda pesada, seguindo uma distribuição como a de Pareto. Este comportamento indica que tamanhos de arquivo muito grandes podem ser encontrados com uma probabilidade não desprezível. Outra conseqüência das distribuições de cauda pesada é a grande variabilidade obtida nas análises.

Para melhorar a análise das cargas da Web, podemos dividir os valores dos tamanhos dos arquivos citados anteriormente em classes. Assim, a utilização dos recursos seria agrupada por categorias, reduzindo a variabilidade e

melhorando os dados estatísticos. Um estudo sobre a variabilidade dos documentos da Web pode ser observado em [4,7].

Dado um conjunto de servidores que hospedam um site em uma única localização, pode-se identificar três arquiteturas principais:

- Arquitetura baseada em cluster: onde os nodos servidores ocultam seus endereços IP na visão dos clientes. O único endereço IP visível é o de um dispositivo (hardware ou software) localizado entre os clientes e o conjunto de servidores, responsável pela distribuição de carga, denominado balanceador de carga. O esquema de um cluster pode ser observado na Figura 8.
- Arquitetura de cluster virtual: A definição deste cluster é a mesma do cluster citada acima, somente com a diferença de que não é utilizado um balanceador. Todos os servidores possuem o mesmo endereço IP e estes são responsáveis pela seleção de qual servidor atenderá a requisição do cliente. A vantagem deste cluster é a remoção de um possível ponto de falha, o balanceador, pois se ele parar de responder no caso do cluster acima, todo o cluster fica sem acesso.
- Arquitetura Web Distribuída: quando os endereços IP reais dos nós servidores são visíveis às aplicações. Esta arquitetura pode ser observada na Figura 9.

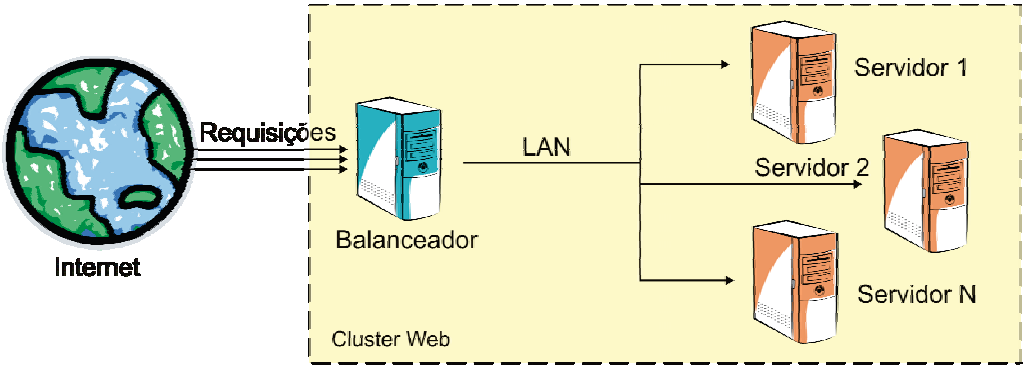

Figura 8 – Arquitetura básica de um cluster Web

Os clusters distribuídos geograficamente são adotados quando o ponto de latência é a rede, o que demanda a localização diferenciada dos servidores ou clusters de servidores na Internet. Esta disposição dos servidores melhoraria a escalabilidade para o atendimento dos acessos. Os clusters podem ser

interconectados por uma rede de alta velocidade, o que facilitaria a troca de informações entre eles.

Na arquitetura distribuída, que foi adotada antes da utilização de clusters, o roteamento é, em geral, decidido pelo sistema DNS. A solução baseada em cluster é mais recente e o roteamento de requisições é inteiramente conduzido pelos componentes internos do cluster, o que proporciona um maior controle na atribuição de tarefas aos servidores, além de mecanismos de segurança e alta disponibilidade. Em geral, a arquitetura distribuída é mais aconselhável para servidores dispersos geograficamente.

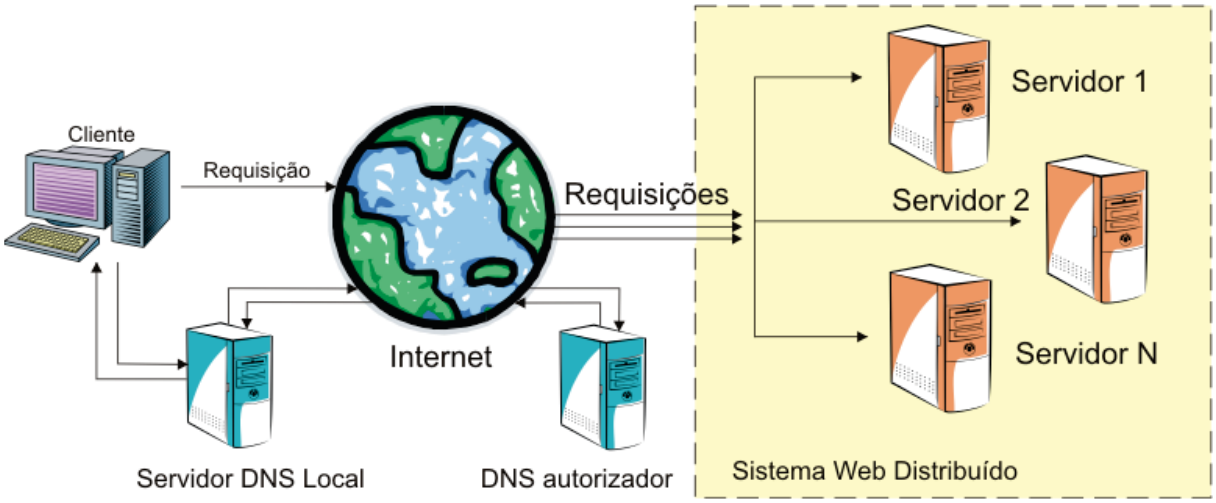

Figura 9 – Cluster Web Distribuído

Embora o cluster possa ser composto por dezenas de nós, somente um nome e um endereço IP é divulgado. Assim, o servidor DNS autorizador executa o mapeamento de um para um, traduzindo o nome no endereço IP de uma entidade dedicada a rotear requisições entre os servidores, formando uma distribuição de carga totalmente transparente aos clientes. Desta forma, esta entidade atua como um distribuidor centralizado que possui um controle bastante fino sobre o escalonamento a ser feito.

Outras formas de selecionar qual o servidor que atenderá a uma requisição podem ocorrer no nível de rede, quando a requisição é direcionada por roteadores ou protocolos de *muticast* ou mesmo no nível do cluster, que pode possuir um tratamento de atendimento que respeite a carga individual de cada um dos servidores, por exemplo.

Para o roteamento das requisições no nível de rede, podem ser utilizados Web switches. Estes switches podem identificar cada um dos servidores individualmente, respeitando seus endereços privados e seus diferentes níveis de protocolo. Estes endereços privados podem ser ou o endereço IP do servidor ou o endereço MAC das placas de rede. Dessa forma, podem ser classificadas de duas formas a utilização destes switches, de acordo com a camada do modelo OSI que eles interagem, normalmente as camadas 4 e 7, respectivamente as camadas de transporte e aplicação.

Os switches de camada 4 são conhecidos pela característica de não serem capazes de interagir com o conteúdo da requisição. Isto ocorre devido o servidor de destino ser contactado no momento em que o primeiro pacote SYN é recebido pelo switch. Como as requisições dos clientes não alcançaram ainda a camada de aplicação, não há a possibilidade de realizar um roteamento levando em consideração as características do conteúdo da requisição.

Os switches de camada 7 podem executar o roteamento de acordo com o conteúdo. Esta característica se deve ao fato de que primeiro o switch realiza o estabelecimento completo da requisição TCP com o cliente, examina o que foi requisitado utilizando o cabeçalho do protocolo HTTP e por fim entrega a requisição ao servidor de destino. Este processo é considerado menos eficiente, mas pode refinar o processo de entrega de pacotes.

De outra forma, o servidor de destino pode responder diretamente para o cliente, chamado de arquitetura *one-way*, ou responder para o switch, que fica encarregado de transportar o processamento da requisição ao cliente, chamado de arquitetura *two-way*. Tipicamente, a arquitetura *one-way* é mais complexa e também mais eficiente, pois o switch processa apenas as informações entrantes, deixando para cargo do servidor encaminhar as respostas aos clientes Web.

2.3.4 Servidores Web mais utilizados

Diversos servidores Web estão disponíveis atualmente. Serão demonstrados os mais utilizados hoje em dia.

O NCSA httpd, idealizado pelo *National Center of Supercomputing Applications,* foi o segundo servidor Web lançado. Foi criado após o primeiro *Web browser* do mundo inventado por Tim Berners-Lee, um cientista do *European*

Organization for Nuclear Research (CERN). O NCSA é o antecessor de diversos servidores Web utilizados atualmente, como o Apache e o Netscape. Muitas características de outros servidores, como controle de acesso e CGI, foram originários do NCSA. O servidor não está sendo mais desenvolvido desde 1998, porém grande parte dos esforços de desenvolvimento foram transferidos para o projeto Apache. Maiores informações sobre o NCSA podem ser obtidas em [23].

O Projeto Apache [25] consiste no esforço conjunto para o desenvolvimento de um servidor HTTP robusto e que atenda às mais exigentes aplicações. O projeto é mantido por diversas pessoas espalhadas pelo mundo trocando idéias e implementações utilizando a Internet, através de um grupo organizado, no intuito de cada vez mais melhorar o servidor criado para ser um software de código aberto e gratuito.

O Apache foi criado usando a versão 1.3 do NCSA HTTPd como base. Todas as correções disponíveis foram aplicadas e adaptações necessárias foram realizadas para possibilitar o lançamento de um novo servidor, o Apache, liberado publicamente pela primeira vez em abril de 1995 na versão 0.6.2. Depois de várias correções, adaptações e adições, a versão 1.0 do servidor foi lançada em dezembro de 1995.

Em apenas quatro meses depois de lançado, ou seja, em abril de 1996, o Apache já era o servidor HTTP mais utilizado ao redor do mundo, de acordo com a pesquisa do *site* NetCraft. O posto de primeiro servidor não foi perdido desde então, mantendo-se hoje em primeiro lugar, instalado em 67,85% dos servidores Web no mundo [24].

Um grupo foi criado em 1999, para manter o servidor, oferecendo suporte organizacional, legal e financeiro ao Apache. Este grupo é conhecido como *Apache Software Foundation* e é responsável por todas as modificações futuras e outras questões relacionadas ao Apache.

O *Internet Information Server* (IIS) da Microsoft é o segundo servidor mais utilizado, estando presente em 21,14% dos servidores [24]. Grande parte do seu sucesso deve-se à sua integração com outros produtos da Microsoft, facilidade de programação de páginas dinâmicas via ASP (*Active Server Pages*) além da facilidade de instalação e configuração. Um problema sério do IIS tem sido suas inúmeras falhas de segurança, que podem permitir desde um ataque de negação de serviço (DOS) até a invasão dos servidores por *hackers*. Maiores informações sobre

o IIS, assim como ajustes para melhoria de desempenho podem ser obtidas em [21]. Outros servidores Web estão disponíveis, como o WebSphere da IBM e o iPlanet da SUN.

Além do servidor em si, muitos softwares adicionais, ou *plug-ins*, são desenvolvidos para adicionar facilidades aos servidores. Um exemplo é a adição do software TomCat no Apache.

O Tomcat, desenvolvido pela Fundação Apache, permite a execução de aplicações no servidor Web Apache. Sua principal característica técnica é estar centrada na linguagem de programação Java, mais especificamente nas tecnologias de Servlets e de Java Server Pages (JSP), permitindo a geração de páginas dinâmicas para este servidor. Esta tecnologia rivaliza com a desenvolvida pela Microsoft, a ASP, que é baseada em Visual Basic.

CAPÍTULO 3
AVALIAÇÃO DE COMPROMISSOS DE PROJETO

Este capítulo descreve duas arquiteturas básicas de projetos de clusters de servidores Web e apresenta os modelos de desempenho, confiabilidade e custo utilizados nas comparações.

3.1 Arquitetura Básica de Clusters de Servidores Web

Ao compor um cluster para executar uma determinada tarefa, temos objetivos específicos tais como obter um determinado tempo de resposta para cada unidade de serviço, prover uma taxa de serviço específica, ou alcançar certos níveis de disponibilidade para o sistema, tudo isso considerando restrições de custo. No entanto, há várias formas de compor um cluster, por exemplo, utilizando máquinas mais ou menos potentes, máquinas que oferecem o melhor desempenho no momento da compra ou ainda máquinas que oferecem a melhor razão custo/desempenho no momento da compra. Neste último tipo, em geral, não estão as máquinas chamadas de topo de linha, mas estão máquinas com bom desempenho. A escolha desta composição pode gerar diferentes combinações de capacidade do cluster, custo, tempo de resposta, disponibilidade e razão custo/desempenho. Esta dissertação discute as relações entre estas métricas na composição de um cluster.

Para apresentar e discutir as relações de interdependência entre os diversos aspectos de desempenho e custo de um cluster vamos definir um modelo básico de composição de clusters e especificar seus parâmetros. O modelo descrito, bem como a metodologia para avaliação dos parâmetros é baseada em [20]. A Figura 10 mostra a arquitetura básica de dois clusters de servidores Web. A Figura 10a mostra um cluster A com n servidores Web idênticos e um balanceamento de carga que distribui igualmente o tráfego total de λ requisições por segundo entre todos os servidores. Cada um dos n servidores do cluster pode atender a X requisições por segundo, de modo que a capacidade total do cluster é de nX requisições por segundo. A Figura 10b mostra um cluster B com m ($m < n$)

servidores Web idênticos, onde cada um dos servidores tem capacidade para executar kX requisições por segundo. Portanto, o cluster B possui menos servidores do que o cluster A, mas a capacidade de cada computador individual do cluster B é maior do que a capacidade de processamento individual dos servidores do cluster A. Nos dois casos, a carga que chega ao sistema é distribuída por um sistema balanceador de carga. Diversas formas de organizar a arquitetura de clusters Web e distribuir a carga são discutidas em [6].

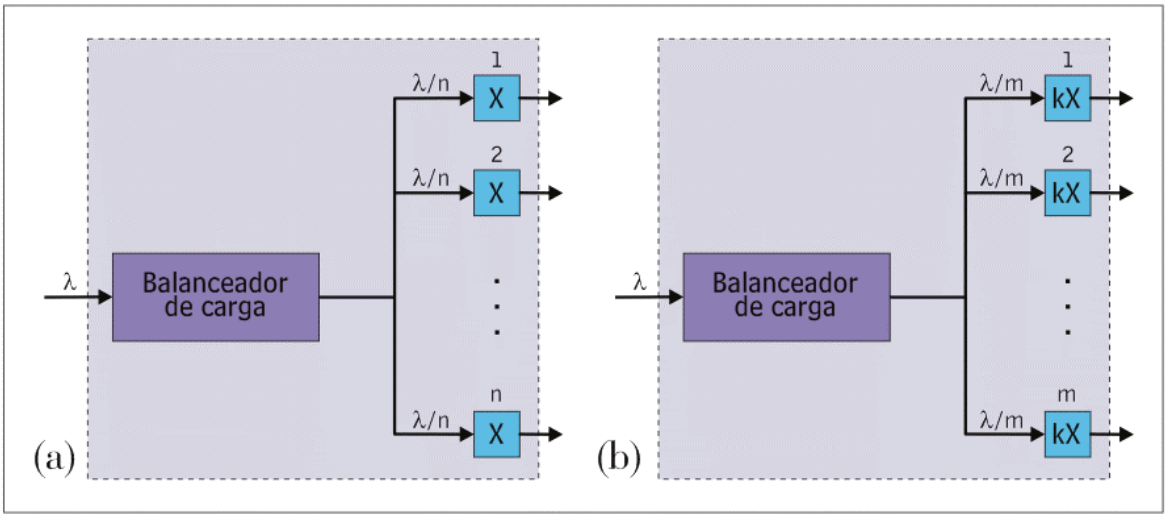

Figura 10 – Arquitetura de dois clusters de servidores Web

O objetivo do trabalho é comparar estes dois modelos de projeto de cluster de acordo com vários critérios, identificando a melhor escolha para cada situação avaliada. Os parâmetros básicos do modelo são demonstrados na Tabela V.

TABELA V
PARÂMETROS DO MODELO

Variável	Significado
n	Número de servidores no cluster A
m	Número de servidores no cluster B
k	Fator de multiplicação para os servidores do cluster B
X	Capacidade média dos servidores do cluster A
λ	Tráfego total

Este objetivo somente pode ser alcançado comparando alguns parâmetros de desempenho entre os clusters. Os parâmetros utilizados neste trabalho são: tempo de resposta médio, capacidade do cluster, custo, confiabilidade e relação custo/desempenho.

Para o tempo de resposta médio, qual deve ser o número de servidores necessários no cluster B para obter o mesmo tempo de resposta médio do cluster A? É necessário analisar diversos fatores, dentre os mais importantes a taxa de chegada das requisições. Quais são as medidas de desempenho que obtemos quando os dois conjuntos têm a mesma capacidade? Para que o custo total dos n servidores do cluster A seja igual ao custo total dos m servidores de B, como irão variar os parâmetros dos clusters para que o custo se mantenha igual? Como manter a mesma confiabilidade nos dois clusters respeitando as medidas de desempenho? Quais serão as medidas de desempenho alcançadas? Como ficam as relações entre os clusters quando o custo/desempenho é levado em consideração?

Estas perguntas podem ser respondidas com as análises que serão feitas para cada um dos casos, onde a partir dessas informações, serão discutidas nas seções seguintes cada um dos cinco casos citados acima buscando identificar qual é a implementação que pode melhor satisfazer uma situação para um determinado projeto de cluster utilizando modelagem analítica.

Estas comparações serão feitas da seguinte forma: para cada critério analisado, analisaremos qual é a composição do cluster B tendo como base a estrutura do cluster A, quando os dois sistemas apresentarem valores iguais para a métrica a ser comparada. Por exemplo, para dois sistemas de mesmo custo, qual seria o tempo de resposta, a taxa de serviço e a disponibilidade de cada sistema. Ou, para dois sistemas com o mesmo tempo de resposta, qual seria o custo, a taxa de serviço e a disponibilidade em cada conjunto.

3.2 Modelos e parâmetros de comparação

Para encontrar o tempo de resposta médio, utilizamos a teoria de filas. A literatura indica que para servidores Web podemos assumir que os processos chegam aos servidores de acordo com os processos de Poisson [26]. No entanto, a distribuição dos tempos de serviço é muito variável [22]. Para clusters que possuem servidores diferentes e o balanceamento de carga não é igualitário, deve-se utilizar o

modelo M/G/m para descrever o sistema, onde G representa uma distribuição genérica, M representa um processo de Markov definindo pela distribuição exponencial e m representa o número de servidores. Porém, o sistema adotado neste trabalho prevê a utilização de servidores iguais e a utilização de um balanceamento igual da carga entre todas as máquinas. Neste contexto, utilizamos o modelo de sistema M/G/1 [15], pois o sistema é tratado como único, onde todas as interações são executadas pelo cluster e não por uma máquina isoladamente. Com a simplificação no modelo e adoção das premissas acima, é possível dizer que nenhuma informação é perdida utilizando o modelo M/G/1. Além disso, há uma comprovação matemática que oferece suporte na equivalência entre estes dois sistemas [15]. Neste contexto, o tempo médio de resposta T para requisições Web é dado por:

$$T = S + \frac{U \times S\left(1 + C^2\right)}{2\left(1 - U\right)} \qquad (1)$$

Nesta equação, que é conhecida como Pollaczek-Khintchine [1], S representa a média do tempo de serviço da requisição, C é o coeficiente de variação do tempo de serviço e U é a utilização de cada servidor, calculada como $\lambda_w S$, onde λ_w é a taxa de chegada média das requisições em cada servidor Web do cluster. A carga total recebida pelo cluster é representada pela variável λ.

Para o cluster A, S = 1/X e $\lambda_w = \lambda/n$. Para o cluster B, S = 1/(kX) e $\lambda_w = \lambda/m$. A teoria de filas pressupõe que os sistemas estão sempre em estado estável, o que significa na prática que a utilização deve assumir valores menores que 1. A partir deste pressuposto, podemos calcular a taxa de chegada máxima possível para ambos os clusters igualando a utilização a 1, obtendo as equações 2 e 3 apresentadas abaixo:

$$U_A = \frac{\lambda}{n} \times \frac{1}{X} < 1 \quad \Rightarrow \quad \lambda < nX \qquad (2)$$

e

$$U_B = \frac{\lambda}{m} \times \frac{1}{kX} < 1 \quad \Rightarrow \quad \lambda < mkX \qquad (3)$$

A equação 2 indica que a taxa de chegada máxima teórica para o cluster A é de nX requisições por segundo e de acordo com a equação 3, a taxa máxima teórica suportada pelo cluster B é de mkX requisições por segundo. Estes valores são teóricos, pois um sistema cuja utilização está próxima de sua capacidade máxima terá possivelmente uma degradação no desempenho devido ao *overhead* gerado pela carga excessiva. Para uma discussão mais completa sobre como o overhead degrada o desempenho nos servidores, consultar a referência [13]. As principais variáveis do modelo estão definidas na Tabela VI.

TABELA VI
DEFINIÇÃO DAS VARIÁVEIS

Variável	Definição
S	Média do tempo de serviço
C	Coeficiente de variação do tempo de serviço
U_x	Utilização do cluster X
T_x	Tempo de resposta do cluster X
N_x	Número médio de requisições processadas no cluster X

Considerando que os dois sistemas recebem a mesma carga, o coeficiente de variação da variável aleatória tempo de serviço (T) é igual em ambos os casos. Os coeficientes de variação das cargas dos servidores Web são altos [22], devido à grande variabilidade observada em T.

O objetivo inicial é encontrar o número m de servidores no cluster B em função das demais variáveis. Assim, substituindo os valores de S e U específicos para cada cluster na equação 1, obtemos as equações 4 e 5 para o tempo médio de resposta para, respectivamente, os clusters A e B:

$$T_A = \frac{1}{X} + \frac{\frac{\lambda}{n} \times \left(\frac{1}{X}\right)^2 \times \left(1 + C^2\right)}{2\left(1 - \frac{\lambda}{n} \times \frac{1}{X}\right)} \qquad (4)$$

e

$$T_B = \frac{1}{kX} + \frac{\frac{\lambda}{m} \times \left(\frac{1}{kX}\right)^2 \times \left(1 + C^2\right)}{2\left(1 - \frac{\lambda}{m} \times \frac{1}{kX}\right)} \qquad (5)$$

Para obter o número médio de requisições no sistema podemos aplicar a Lei de Little. De acordo com esta lei, o número médio de requisições processadas é $N_A = \lambda \times T_A$, para o cluster A, e $N_B = \lambda \times T_B$, para o cluster B.

Nas próximas seções serão comparados os clusters A e B quanto às seguintes métricas: tempo de resposta, capacidade, custo, confiabilidade e custo/desempenho. Em cada uma das comparações, o valor de cada métrica é igualado nos dois sistemas e o efeito é então avaliado nas demais métricas.

3.3 Equiparando tempos de resposta

O objetivo desta seção é verificar, para uma dada composição do cluster A, qual será a composição do cluster B, mais especificamente o valor da variável m, para que os dois sistemas tenham o mesmo tempo de resposta. Além disso, verificamos também os valores das demais métricas para as duas comparações quando o tempo de resposta é igual para ambas. Para determinar o valor de m que iguala o tempo de resposta utilizando o mesmo valor de λ, foram igualadas as equações 4 e 5, que foram manipuladas algebricamente da seguinte forma:

$$T_A = T_B$$

$$\frac{1}{kX} + \frac{\frac{\lambda}{m} \times \left(\frac{1}{kX}\right)^2 \times \left(1 + C^2\right)}{2\left(1 - \frac{\lambda}{m} \times \frac{1}{kX}\right)} = \frac{1}{X} + \frac{\frac{\lambda}{n} \times \left(\frac{1}{X}\right)^2 \times \left(1 + C^2\right)}{2\left(1 - \frac{\lambda}{n} \times \frac{1}{X}\right)}$$

Inicialmente, retiramos as equações que estavam elevadas ao quadrado e passamos a fração 1/kX para o outro lado.

$$\frac{\frac{\lambda}{m}\times\frac{1}{k^2}\times\frac{1}{X^2}\times\left(1+C^2\right)}{2\left(1-\frac{\lambda}{mkX}\right)}=\frac{1}{X}-\frac{1}{kX}+\frac{\frac{\lambda}{n}\times\frac{1}{X^2}\times\left(1+C^2\right)}{2\left(1-\frac{\lambda}{nX}\right)}$$

Realizamos a multiplicação nos denominadores das frações e a soma entre 1/X e –1/kX.

$$\frac{\frac{\lambda}{mk^2X^2}\times\left(1+C^2\right)}{\frac{2mkX-2\lambda}{mkX}}=\frac{k-1}{kX}+\frac{\frac{\lambda}{nX^2}\times\left(1+C^2\right)}{\frac{2nX-2\lambda}{nX}}$$

Executamos a divisão das duas frações principais.

$$\frac{\frac{\lambda}{mk^2X^2}\times\left(1+C^2\right)\times mkX}{2mkX-2\lambda}=\frac{k-1}{kX}+\frac{\frac{\lambda}{nX^2}\times\left(1+C^2\right)\times nX}{2nX-2\lambda}$$

Simplificamos as equações, eliminando os valores multiplicadores.

$$\frac{\frac{\lambda}{kX}\times\left(1+C^2\right)}{2\left(mkX-\lambda\right)}=\frac{k-1}{kX}+\frac{\frac{\lambda}{X}\times\left(1+C^2\right)}{2\left(nX-\lambda\right)}$$

Multiplicamos a equação por 2.

$$\frac{\frac{\lambda}{kX}\times\left(1+C^2\right)}{mkX-\lambda}=2\left[\frac{\left(k-1\right)}{kX}+\frac{\frac{\lambda}{X}\times\left(1+C^2\right)}{2\left(nX-\lambda\right)}\right]$$

Multiplicamos as frações da direita por 2.

$$\frac{\frac{\lambda}{kX}\times\left(1+C^2\right)}{mkX-\lambda}=\frac{2(k-1)}{kX}+2\left[\frac{\frac{\lambda}{X}\times\left(1+C^2\right)}{2(nX-\lambda)}\right]$$

Passamos o numerador da equação à esquerda para a direita como um fator divisor.

$$\frac{1}{mkX-\lambda}=\left(\frac{kX}{\lambda\left(1+C^2\right)}\right)\left[\frac{2(k-1)}{kX}+\frac{\frac{\lambda}{X}\times\left(1+C^2\right)}{(nX-\lambda)}\right]$$

Executamos a multiplicação entre as frações do lado direito da equação.

$$\frac{1}{mkX-\lambda}=\frac{2(k-1)\times kX}{kX\times\lambda\times\left(1+C^2\right)}+\frac{\frac{\lambda}{X}\times\left(1+C^2\right)\times kX}{(nX-\lambda)\times\lambda\times\left(1+C^2\right)}$$

Eliminamos variáveis.

$$\frac{1}{mkX-\lambda}=\frac{2(k-1)}{\lambda\times\left(1+C^2\right)}+\frac{k\lambda}{(nX-\lambda)\times\lambda}$$

Elevamos ambos os lados da equação por –1, para inverter as frações.

$$\left(\frac{1}{mkX-\lambda}\right)^{-1}=\left(\frac{2(k-1)}{\lambda\times\left(1+C^2\right)}+\frac{k}{nX-\lambda}\right)^{-1}$$

Aqui está a equação aplicando a operação de potência.

$$mkX - \lambda = \frac{1}{\frac{2(k-1)}{\lambda \times (1+C^2)} + \frac{k}{nX - \lambda}}$$

Isolando o valor de mkX.

$$mkX = \lambda + \frac{1}{\frac{2(k-1)}{\lambda \times (1+C^2)} + \frac{k}{nX - \lambda}}$$

Após estas manipulações, a seguinte fórmula é obtida:

$$m = \frac{1}{kX}\left[\lambda + \frac{1}{\frac{2(k-1)}{\lambda \times (1+C^2)} + \frac{k}{nX - \lambda}}\right] \qquad (6)$$

De acordo com a fórmula obtida, para calcularmos o número total de servidores no cluster B para que se tenha o mesmo tempo de resposta no cluster A, os seguintes parâmetros são necessários:

- Tráfego total λ
- Capacidade média X dos servidores no cluster A
- Número n de servidores no cluster A
- Fator de multiplicação k da capacidade de processamento dos servidores no cluster B em relação aos servidores do cluster A
- Coeficiente de variação C do tempo de serviço.

A equação 6 também indica que quando λ cresce e se aproxima de nX requisições por segundo (valor máximo que pode ser assumido), m tende a ter seu valor igual a n/k. Quando este valor é alcançado, ambos os clusters tem a mesma capacidade de processamento. É importante lembrar que esta capacidade de processamento que está sendo implementada aqui não representa a capacidade de um componente em si, como processador, mas sim do sistema como um todo.

Para que se fosse possível encontrar valores reais para as variáveis utilizadas, foram consultados benchmarks reais na Internet. Valores encontrados nos benchmarks Specweb99 [32] e TPC-W [34] são listados nas Tabelas a seguir.

TABELA VII
BENCHMARK SPECWEB99

Servidor	Data do Teste	Valor obtido
Dell PowerEdge 2400/66	Março/2000	732
Dell PowerEdge 8450/700	Novembro/2000	7.500
IBM eServer xSeries 370	Fevereiro/2001	2.700
Compaq Proliant DL320	Janeiro/2002	1.943
IBM pSeries 660 6M1	Março/2002	10.000
IBM eServer xSeries 345	Julho/2002	5.000

O Specweb99 é um benchmark cliente/servidor que verifica o número máximo de conexões simultâneas que um servidor Web é capaz de suportar. A carga do teste é gerada por um software instalado em máquinas-clientes interligadas ao servidor testado. Os servidores mostrados na tabela acima possuem configurações bastante distintas. Porém, a motivação é demonstrar quais parâmetros podem ser utilizados nas variáveis do modelo aqui apresentado. Maiores detalhamentos sobre as configurações dos servidores podem ser obtidas em [32].

Para os servidores testados, é possível perceber a diferença nos valores obtidos nos testes. Para os dois modelos de servidores da Dell, há uma diferença maior que 10 vezes no número de conexões atendidas. Entre os dois servidores IBM do ano de 2002, os valores demonstram que o modelo eServer xSeries 345 é capaz de atender o dobro de conexões do que o modelo pSeries 660. Comparando-se o IBM pSeries e o servidor Compaq Proliant, a diferença nos resultados é mais que 5 vezes maior.

TABELA VIII
BENCHMARK TPC-W

Servidor	Data da Submissão	Valor obtido
IBM e(logo) xSeries 350	17/12/2001	7.073
Dell PowerEdge 6400/900	19/12/2001	6.622
Dell PowerEdge 6400/900	28/01/2002	7.783
Dell PowerEdge 6650	22/08/2002	10.449
IBM eServer xSeries 440	12/09/2002	21.139

O benchmark TPC-W faz uma simulação no atendimento em sites de comércio eletrônico. Os usuários são emulados utilizando os mesmos princípios de consumidores reais navegando pelo site, gerando também uma carga equivalente. A métrica principal calculada pelo TPC-W é chamada de WIPS. WIPS é o número de interações Web por segundo que o sistema é capaz de atender. Os valores da Tabela acima demonstram os resultados para alguns servidores testados pelo TPC-W. É possível perceber nos servidores selecionados há uma diferença na capacidade de atendimento em torno de 3 vezes entre os servidores Dell PowerEdge 6400 e o IBM eServer xSeries 440. O servidor Dell PowerEdge 6400 aparece duas vezes na listagem por ter seus resultados submetidos novamente após alterações no servidor. Maiores informações podem ser obtidas em [34].

A Figura 11 mostra o valor de m como função de λ para k = 2, C = 5, X = 100 e para três valores de n (5, 10 e 15). Os valores utilizados seguem as características encontradas no benchmarks apresentados anteriormente, onde o coeficiente de variação C é capaz de representar a variabilidade de um servidor Web tradicional. Somente neste primeiro exemplo o valor adotado para k é igual a 2. Este valor foi escolhido pois demonstra mais facilmente a diferença entre as curvas geradas, onde estas alcançam seu valor máximo quando a taxa de chegada atinge o valor limite de acordo com as equações 4 e 5. Quando o cluster A tem 15 servidores e recebe 900 requisições por segundo, o cluster B precisa de 6 servidores para obter o mesmo tempo de resposta médio de 0,2 segundos. Neste ponto os servidores do cluster A estão 60% utilizados, enquanto que no cluster B a utilização chega a 75%. Neste caso o valor de k foi igualado a 2 para facilitar a observação das conclusões na Figura 11.

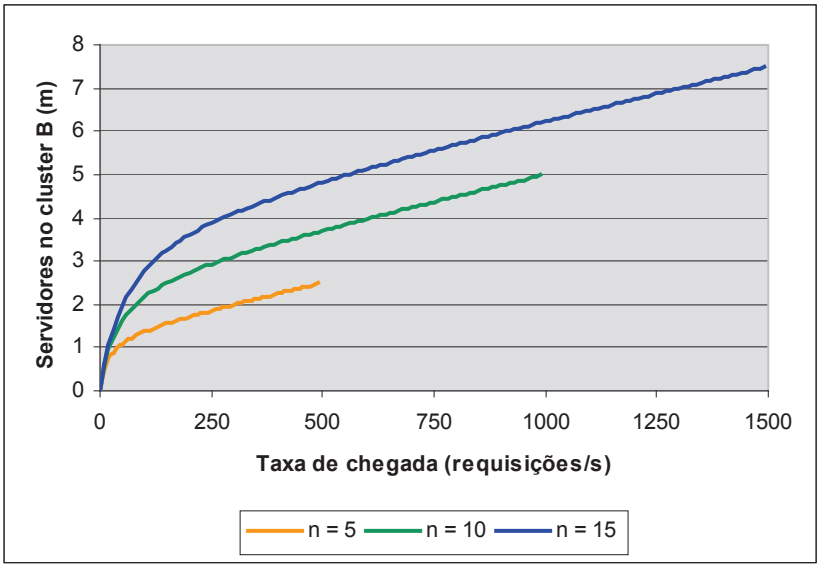

Figura 11 – Servidores no cluster B para o caso do mesmo tempo de resposta

A Figura 11 e a equação 6 mostram que para uma taxa de chegada considerada razoável, m cresce linearmente com λ, na taxa de $(k-1)/(k^2X)$. A Figura acima mostra que esta taxa de crescimento não depende de n.

A Tabela XIX mostra o comportamento das variáveis para o caso do mesmo tempo de resposta. Quando ambos os clusters recebem 900 requisições por segundo, o cluster A está com sua capacidade 60% utilizada para os três exemplos. Para o primeiro caso, quando o valor de n é igual a 5, o cluster B não tem a capacidade de atender a demanda de requisições. Quando n é igual a 10, os clusters respondem as requisições em 0,59 segundos em média. O cluster B utiliza 90% de seus recursos possuindo 5 máquinas. Para o terceiro caso, o tempo de resposta esperado é de 0,2 segundos. O cluster A possui 15 máquinas e o cluster B possui 6 máquinas. A utilização do cluster B atinge 75%. No último caso, quando n é igual a 25, o cluster B possui 9 máquinas e uma menor utilização em relação ao cluster A. Percebe-se que quando o valor de n é incrementado, o cluster B tende a diminuir sua utilização. Sendo assim, para valores de n pequenos, o melhor projeto é o apresentado para o cluster A, onde obtemos melhor tempo de resposta. Caso contrário, quando n aumenta, o cluster B assume o melhor tempo de resposta.

TABELA XIX
VARIÁVEIS PARA O CASO DO MESMO TEMPO DE RESPOSTA

λ	K	C	m	n	T_A=T_B (s)	U_A	U_B
900	2	5	–	5	–	60%	> 100%
900	2	5	5	10	0,59	60%	90%
900	2	5	6	15	0,2	60%	75%
900	2	5	9	25	0,07	60%	50%

3.4 Equiparando capacidades

Quando os dois clusters têm a mesma capacidade de processamento, então nX = mkX e, portanto, m = n/k. Uma vez que a capacidade e a taxa de chegada das requisições são iguais para ambos os clusters, a utilização também é a mesma. Substituindo m = n/k nas equações 4 e 5, obtêm-se que $T_B = T_A/k$ e $N_B = N_A/k$, o que está de acordo com a Lei de Little. Assim, é possível dizer que, para qualquer valor de λ , o tempo médio de resposta do cluster B é k vezes menor do que o do cluster A e o número médio de requisições em atendimento em B é k vezes menor do que o número médio de requisições no cluster A, quando os clusters tem a mesma capacidade.

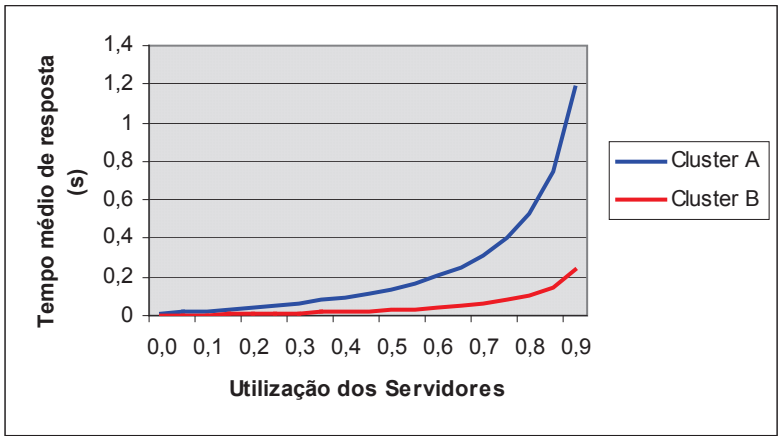

Figura 12 – Média do tempo de resposta em relação à utilização dos servidores para capacidades equivalentes

A Figura 12 mostra a variação no tempo de resposta médio em função da utilização dos servidores em cada cluster para k = 5, C = 5 e X = 100, quando

ambos os clusters têm a mesma capacidade. Podemos observar que o tempo médio de resposta do cluster B é sempre cinco vezes menor do que no cluster A. Isto ocorre devido ao valor da variável k utilizada, que é igual a 5.

A Tabela X mostra vários exemplos para o caso da mesma capacidade. Com a mudança das variáveis, o comportamento global do sistema se mantém. Neste caso, a melhor escolha é o cluster B, que mantém sempre o tempo de resposta k vezes menor.

TABELA X
VARIÁVEIS PARA O CASO DA MESMA CAPACIDADE

λ	k	C	X	m	n	T_A(s)	T_B (s)	$U_A = U_B$
900	5	5	100	3	15	0,205	0,041	60%
900	5	5	100	4	20	0,1163	0,0232	45%
900	5	5	100	6	30	0,0657	0,0131	30%
1200	5	5	100	4	20	0,205	0,041	60%

3.5 Equiparando custos

Uma vez que o custo é um ponto chave nas implementações dos clusters, então esta comparação é de suma importância para avaliarmos a melhor opção em relação a uma métrica essencial de comparação de sistemas, que é a relação custo/desempenho.

O custo total do cluster A, com seus n servidores, deve ser o mesmo do cluster B com m servidores. Considerando C(X) o custo de um servidor com capacidade X, então n C(X) deve ser igual a m C(kX). Os clusters terão o mesmo custo quando m = n C(x) / C(kX). Se o custo do servidor for proporcional a sua capacidade, isto é, C(kX) = k. C(X), então m = n/k, e custos iguais também significam capacidades iguais. Neste caso, todas as discussões que foram tomadas para o caso da mesma capacidade também estariam válidas aqui, pois os parâmetros encontrados para a construção dos clusters foram exatamente os mesmos.

No entanto, as variações de custo nem sempre são diretamente proporcionais à capacidade dos sistemas. As funções de custo podem ser sublineares ou superlineares. Para funções de custo sublineares há uma economia

na escala quando a capacidade do servidor aumenta, isto é, o custo por unidade de capacidade diminui quando a capacidade aumenta. No custo superlinear, o custo por unidade de capacidade do servidor aumenta com o aumento da capacidade.

Por exemplo, para C(X) = $\alpha\sqrt{x}$ (α > 0), então m = n / $\sqrt{k}$. Usando estes valores para m na equação 5, obtém-se a média do tempo de resposta para o cluster B. Verificando-se as equações 2 e 3, é possível calcular a utilização dos servidores no cluster A, que é $\sqrt{k}$ vezes a utilização dos servidores no cluster B.

A Figura 13 mostra que o tempo médio de resposta no cluster A cresce muito mais rapidamente que no cluster B para k = 5, X = 100 e C = 5. Quando o cluster recebe 1.200 requisições por segundo, por exemplo, um servidor no cluster A tem a utilização de 80% e outro servidor no cluster B está com sua utilização em 34%. O tempo de resposta médio é de 0,53 segundos para o cluster A e é aproximadamente 32 vezes maior do que o observado no cluster B.

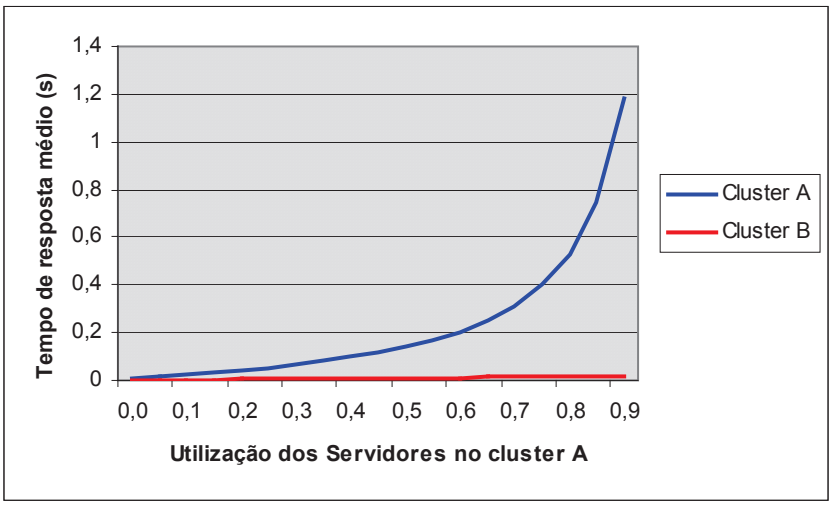

Figura 13 – Tempo de resposta médio em função da utilização no cluster A

A Tabela XI demonstra como as variáveis alteram a composição dos clusters e seus tempos de resposta para o primeiro caso do custo. A variável α é igual a 100, ou seja, cada conjunto de requisições pode ser comprado por um valor igual a US$ 100. Com o aumento do número de servidores em A, a comparação dos custos mantém-se com resultados equivalentes. Mesmo com as alterações, o cluster B apresenta-se melhor, pois com o mesmo custo, ele fica com sua utilização e seu tempo de resposta menor do que os apresentados pelo cluster A.

TABELA XI
VARIÁVEIS PARA O CASO DO MESMO CUSTO – FUNÇÃO RAIZ QUADRADA

λ	k	C	α	m	n	T_A(s)	T_B (s)	U_A	U_B
900	5	5	100	7	15	0,205	0,0115	60%	25,7%
1200	5	5	100	9	20	0,205	0,0115	60%	26,6%
900	5	5	100	9	20	0,1163	0,0085	45%	20%
900	5	5	100	14	30	0,0657	0,0060	30%	12,8%

Se a função de custo considerada for uma função quadrática, utiliza-se a função no formato $C(X) = \alpha X^2$ ($\alpha > 0$). Desta forma tem-se que $m = n/k^2$. As equações 2 e 3 demonstram que a utilização nos servidores do cluster B é k vezes a utilização dos servidores do cluster A.

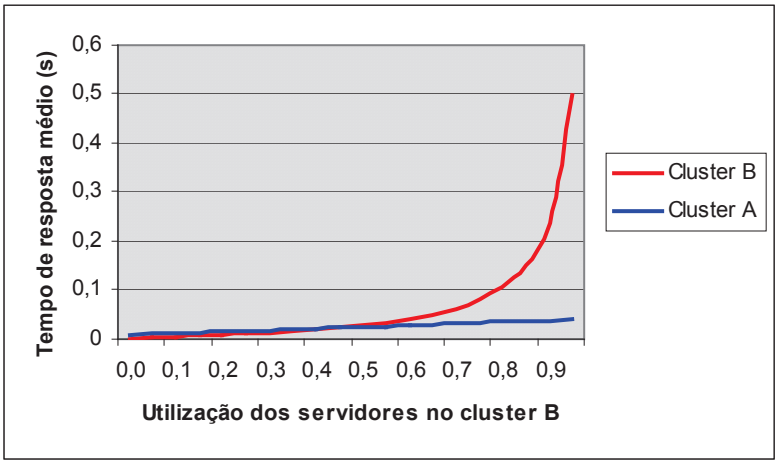

Figura 14 – Tempo de resposta médio em função da utilização no cluster B

A Figura 14 apresenta o tempo de resposta para ambos os clusters utilizando a função quadrática para k = 5, X = 100 e C = 5. Para pouco tráfego, o cluster B apresenta menor tempo de resposta que o cluster A devido a pouca fila encontrada no cluster B, além de seus servidores terem um desempenho k vezes maior. Entretanto, quando a carga aumenta, o cluster B apresenta um grande aumento no tempo de resposta, enquanto que o cluster A tem um acréscimo mais lento. Pode-se verificar então que, para um alto tráfego, o cluster B é melhor quando se utiliza a função raiz quadrada, mas o inverso é verdadeiro para a função quadrática.

A Tabela XII mostra os resultados para o caso do custo função quadrática. Em todos os exemplos apresentados abaixo o cluster B apresenta a pior utilização devido o pequeno número de servidores necessários para manter o custo igual utilizando a função quadrática. Para este caso, a melhor escolha é o cluster A, que mantém um menor tempo de resposta e uma menor utilização. É importante lembrar também que para utilizações baixas, o cluster B pode apresentar melhores resultados.

TABELA XII
VARIÁVEIS PARA O CASO DO MESMO CUSTO – FUNÇÃO QUADRÁTICA

λ	k	C	α	m	n	T_A(s)	T_B (s)	U_A	U_B
900	5	5	100	1	15	0,205	–	60%	> 100%
900	5	5	100	1	25	0,08	–	35%	> 100%
900	5	5	100	2	35	0,054	0,236	25,7%	90%

3.6 Equiparando confiabilidade

Com o aumento da complexidade da Web, diminuição nos tempos para desenvolvimento dos sistemas e incremento nas tentativas de invasão, questões como confiabilidade, segurança e disponibilidade tornaram-se discussões primordiais no projeto dos sistemas.

A confiabilidade é a probabilidade de um sistema ou de um de seus componentes operar de forma correta e contínua em um determinado período de tempo. Define-se então r_A como a confiabilidade de cada servidor no cluster A e r_B como a confiabilidade de cada servidor do cluster B.

A confiabilidade de cada um dos clusters pode ser medida de acordo com as fórmulas abaixo, com mostrada no capítulo anterior e em [20]:

$$R_A = 1 - (1 - r_A)^n \qquad (7)$$

e

$$R_B = 1 - (1 - r_B)^m \qquad (8)$$

Igualando as equações apresentadas acima e aplicando-se logaritmos, chegamos na seguinte equação que calcula o valor de m:

$$m = \frac{n \log(1 - r_A)}{\log(1 - r_B)} \qquad (9)$$

Quando a confiabilidade é igual em ambos os clusters, então o valor de m seguirá a fórmula acima. Se as máquinas dos clusters tivessem a mesma confiabilidade, seguindo a equação 9, o valor de m seria exatamente o valor de n, ou seja, ambos os clusters deveriam ter o mesmo número de máquinas para manter a mesma confiabilidade. Este resultado é baseado na confiabilidade dos sistemas em paralelo, onde cada máquina não irá interferir na outra, porém quanto mais máquinas, mais confiável é o sistema. Para este caso de comparação utilizamos confiabilidades diferentes para cada cluster. Como o cluster A possui mais máquinas, a indisponibilidade de uma delas ocasionaria uma menor perda de confiabilidade se uma máquina no cluster B parasse de funcionar. Sendo assim, foram escolhidos valores diferenciados para cada cluster.

Comparando o tempo de resposta médio quando a confiabilidade é de 0,9 para cada servidor do cluster A e de 0,999 para os servidores do cluster B e k = 5, X = 100 e C = 5, serão necessários 5 servidores no cluster B para se ter a mesma confiabilidade do cluster A. Os valores de confiabilidade utilizados no cálculo de m para cada um dos clusters consideram que o cluster A possui mais servidores que o cluster B. Portanto, a confiabilidade esperada de cada servidor do cluster A individualmente pode ser menor do que os servidores do cluster B, haja visto que como o cluster B possui menos máquinas, se uma falhar isto terá um impacto maior do que se uma máquina no cluster A ter seu funcionamento interrompido.

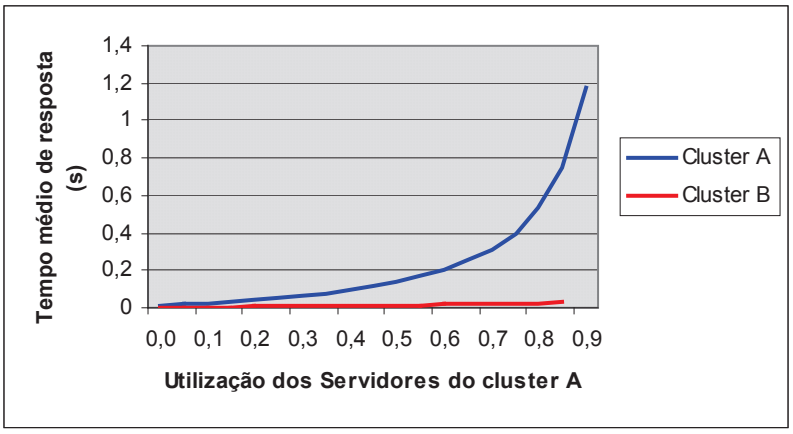

Figura 15 – Tempo de resposta médio em função da utilização no cluster A.

A Figura 15 mostra o tempo de resposta médio versus a utilização dos servidores no cluster A. O cluster B tem um tempo de resposta significativamente menor que o encontrado no cluster A. Quando o cluster A está com sua utilização em 80%, por exemplo, a utilização do cluster B é de 48%. Com um tempo de resposta de B em 0,026 segundos, o tempo de resposta para A é cerca de 20 vezes maior. Para a Figura acima, foi utilizada a utilização do cluster A devido a utilização do cluster ter atingido 100% antes que a utilização do cluster B alcançasse este valor.

A Tabela XIII contém os valores obtidos nas comparações para o caso da mesma confiabilidade. Para todos os valores a utilização do cluster B, assim como seu tempo de resposta encontrado, são menores do que os do cluster A. Além disso, quanto maior a carga que está sendo enviada aos clusters, a diferença entre os clusters tende a aumentar favorecendo o cluster B. Portanto, para este caso o melhor cluster é o cluster B.

TABELA XIII
VARIÁVEIS PARA O CASO DA MESMA CONFIABILIDADE

λ	k	C	m	n	T_A(s)	T_B (s)	U_A	U_B
900	5	5	5	15	0,205	0,016	60%	36%
900	5	5	7	20	0,116	0,011	45%	25,7%
900	5	5	10	30	0,065	0,007	30%	18%
1200	5	5	5	15	0,53	0,026	80%	48%

3.7 Relação Custo/Desempenho

A relação custo/desempenho é uma métrica que visa comparar dois ou mais sistemas incluindo no custo valores de licenciamento de *software* e *hardware*, instalação e manutenção dos sistemas por um determinado tempo estipulado [13]. O desempenho é medido, por exemplo, em termos de *throughput* em relação a uma dada constante. Um dos casos que pode ser analisado por essa métrica ocorre quando se quer saber quanto um determinado valor pode comprar por número de requisições executadas.

Para avaliar a relação custo/desempenho consideramos que os dois clusters têm a mesma capacidade. Os parâmetros dos clusters seguem os utilizados no caso de mesma capacidade mostrado na seção 3.6. Aplicamos os modelos de custo considerando a constante α igual a 100, onde o valor de α representa o valor encontrado por unidade de capacidade. As Figuras 16 e 17 apresentam os gráficos com a relação custo/desempenho para os clusters A e B para as funções de custo raiz quadrada e quadrática respectivamente. Foi utilizada escala logarítmica para o eixo y. Podemos observar que para o primeiro caso, estes dados demonstram que o cluster B apresenta a melhor relação custo/desempenho, para a função de custo raiz quadrada. Para o segundo caso, o cluster A apresenta a melhor relação custo/desempenho.

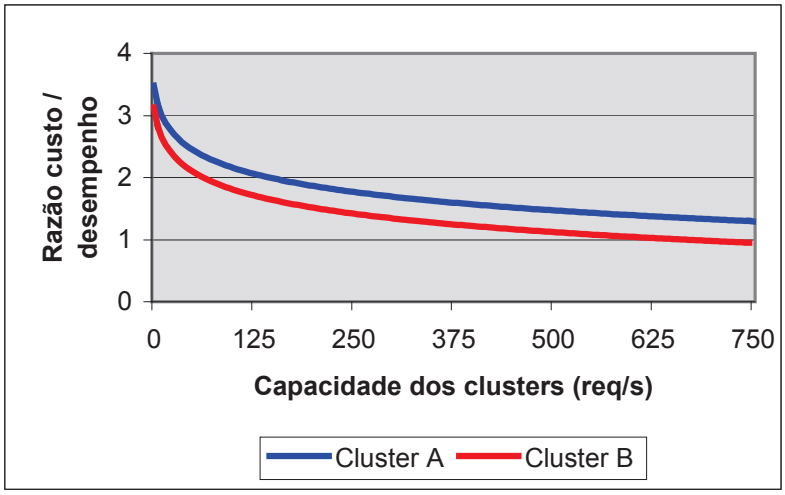

Figura 16 – Razão custo/desempenho em relação à capacidade dos clusters A e B utilizando a função custo raiz quadrada.

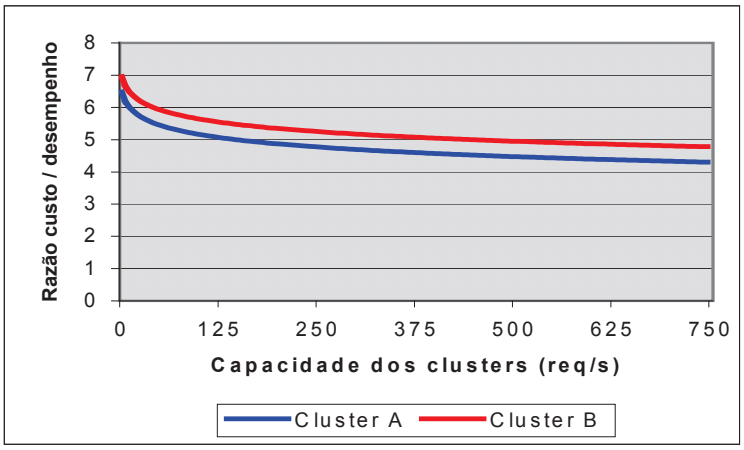

Figura 17 – Razão custo/desempenho em relação à capacidade dos clusters A e B utilizando a função custo quadrática.

3.8 Síntese das comparações

A Tabela XIV mostra o tempo médio de resposta para o cluster B, o número de servidores no cluster e a utilização de cada servidor para os cinco casos de comparação apresentados neste trabalho considerando-se $\lambda = 900$ requisições por segundo. Esta comparação considera a utilização de 60% da capacidade e um tempo de resposta de 0,205 segundos para o cluster A.

A Tabela mostra qual deve ser o valor de m para cada critério utilizado para comparar os clusters, bem como o tempo de resposta que será obtido para a composição, e que deve ser comparado com o tempo médio de resposta de 0,205 segundos para o cluster A. Para o caso do custo igual utilizando uma função quadrática, a utilização do cluster B excede 100% e o tempo de resposta tende ao infinito. Isto indica que, para manter o mesmo custo, o cluster B teria apenas uma máquina, que sozinha não seria capaz de processar a carga especificada para esta comparação.

Em alguns casos apresentados não foi possível obter o número exato de servidores no cluster B para satisfazer a quantidade de máquinas requisitadas naquele caso. Um exemplo é o caso do mesmo tempo de resposta, onde apesar de no cluster A este tempo ser de 0,205s, o valor encontrado para o cluster B é de 0,236s, devido ao arredondamento no número de servidores.

TABELA XIV
TAMANHO (M) REQUERIDO PARA O CLUSTER B PARA λ = 900 REQUISIÇÕES/SEGUNDO EM VÁRIOS CASOS

T_B (s)	m	U_B (%)	Caso
0,236	2	90	Igual tempo de resposta
0,041	3	60	Igual capacidade total
0,0115	7	26	Igual custo: $C(x) = \sqrt{x}$
∞	1	> 100	Igual custo: $C(x) = x^2$
0,017	5	36	Igual confiabilidade do cluster

É possível perceber na tabela que de acordo com a métrica que foi escolhida para comparação, os valores obtidos nas demais métricas tiveram grandes alterações. Nenhum dos clusters foi o melhor em todos os casos de comparação. Desta forma, a melhor escolha deve ser baseada na métrica que melhor se adapta às necessidades do cliente que irá utilizar o cluster.

CAPÍTULO 4
ESTUDOS DE CASO

Neste capítulo serão discutidos dois estudos de caso. Será possível analisar o comportamento destes clusters de diferentes tamanhos e capacidades comparando os resultados.

Na seção 4.1 é demonstrado um exemplo para clusters de grande porte, como o do Google. Na seção 4.2 são comparados clusters com alto poder de processamento.

4.1 Comparando clusters de grande porte

Nesta seção comparamos dois clusters de grande capacidade. Os dados utilizados como parâmetro dos modelos foram obtidos a partir de informações do cluster do site de buscas Google [5,27]. Estes dados indicam que o Google tem entre 15.000 e 20.000 máquinas, que são computadores comuns do tipo PC. O critério básico de compra é a melhor relação custo/desempenho e não o desempenho máximo por máquina. Além disso, as taxas de requisições são muito altas e o tempo médio de resposta está em torno de 0,20s [14]. Considerando ainda os tempos de processamento relacionados à tarefa de pesquisa do Google e a grande quantidade e diversidade de buscas realizadas nos seus índices, vamos considerar também um maior coeficiente de variação, sabendo-se também que o padrão de carga da Internet é considerado de cauda pesada [3, 7, 9]. Assim, as variáveis serão configuradas da seguinte forma: n = 20.000, C = 15 e X = 200, definindo o cluster C. O cluster D conterá máquinas cujo poder de processamento é k vezes maior. Inicialmente k = 5. Se algum valor diferente for utilizado, o mesmo será informado.

A Figura 18 mostra o valor de m para k = 2 e demais valores iguais aos citados acima em função de λ para os valores de n iguais a 5.000, 10.000 e 20.000. Como no exemplo anterior, as curvas têm seus limites de acordo com as equações 4 e 5. Neste caso, quando o cluster C, com 20.000 servidores e utilização de 35%,

recebe 1.400.000 (caso 1) requisições por segundo, são necessários 6.724 servidores no cluster D para que este fique com a utilização em 52% e o tempo de resposta em 0,309 segundos. Se o número de requisições for igual a 2.400.000 (caso 2), o cluster C fica com sua utilização em 60% e são necessários 7.994 servidores no cluster D para que ele fique 85,2% utilizado.

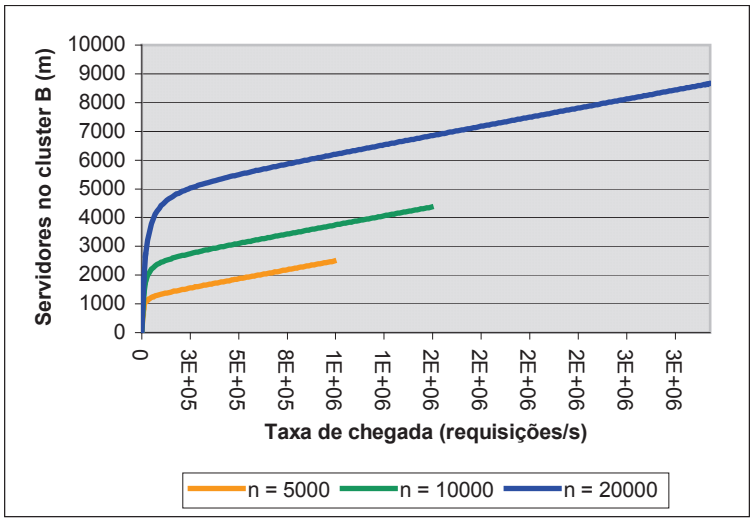

Figura 18 – Servidores no cluster D para o caso do mesmo tempo de resposta

A Tabela XV mostra diversos casos para o mesmo tempo de resposta. Em todos os casos apresentados abaixo, o cluster C está com a utilização em um nível menor, indicando também um menor overhead gerado pelo excesso de carga em sistemas reais. Pode ser observado também que quando o fator de multiplicação k diminui, a utilização do cluster D tende a se aproximar a do cluster C.

TABELA XV
VARIÁVEIS PARA O CASO DO MESMO TEMPO DE RESPOSTA

λ	k	C	m	n	$T_C=T_D$ (s)	U_C	U_D
2.000.00	5	15	2.397	20.000	0,570	50%	83,43%
1.800.00	5	15	2.236	20.000	0,467	45%	80,5%
1.800.00	2	15	7.235	20.000	0,467	45%	62,19%
3.000.00	5	15	3.200	20.000	1,696	75%	93,75%

No caso de mesma capacidade, como demonstrado na Figura 19, como a variável k continua com o mesmo valor igual a 5, então a relação entre os tempos de resposta dos clusters C e D variarão segundo esta variável. Os valores

de tempo de resposta para o cluster C serão sempre 5 vezes maiores do que os encontrados no cluster D.

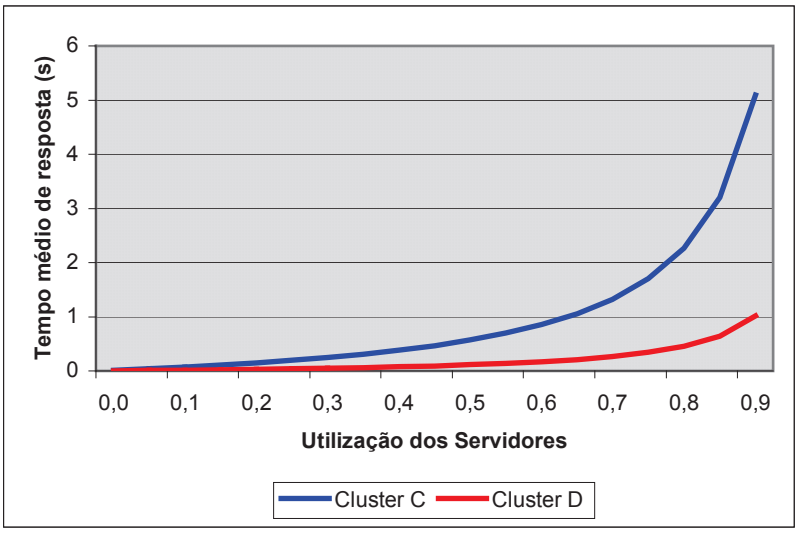

Figura 19 – Tempo médio de resposta em relação à utilização dos servidores para capacidades equivalentes

A Tabela XVI possui alguns valores de referência para o caso da mesma capacidade dos clusters C e D. Como já foi mencionado anteriormente, para este caso o tempo de resposta entre os clusters terá variação de acordo com o valor da variável k. Sendo assim, a melhor escolha é pelo cluster que possui o menor tempo de resposta, ou seja, o cluster D.

TABELA XVI
VARIÁVEIS PARA O CASO DA MESMA CAPACIDADE

λ	k	C	X	m	n	T_C(s)	T_D (s)	$U_C = U_D$
2.400.00	5	15	200	4.000	20.000	0,852	0,170	60%
3.200.00	5	15	200	4.000	20.000	2,265	0,453	80%
2.400.00	10	15	200	2.000	20.000	0,852	0,085	60%
1.200.00	5	15	200	4.000	20.000	0,247	0,494	30%

Na Figura 20, para o caso de mesmo custo com a função de custo de raiz quadrada, quando o cluster recebe o mesmo fluxo de informações do caso 1, o cluster C fica com a utilização em 35% e o cluster D fica 15,65% utilizado com 8.945 servidores. O tempo médio de resposta é de 0,022 segundos para o cluster D, que é aproximadamente 14 vezes maior do que o encontrado no cluster C. Para o caso 2,

a utilização no cluster C fica em 60% e no cluster D a utilização é de 26,8%. O tempo de resposta de C é 0,8525 segundos e no cluster D, 0,0424 segundos.

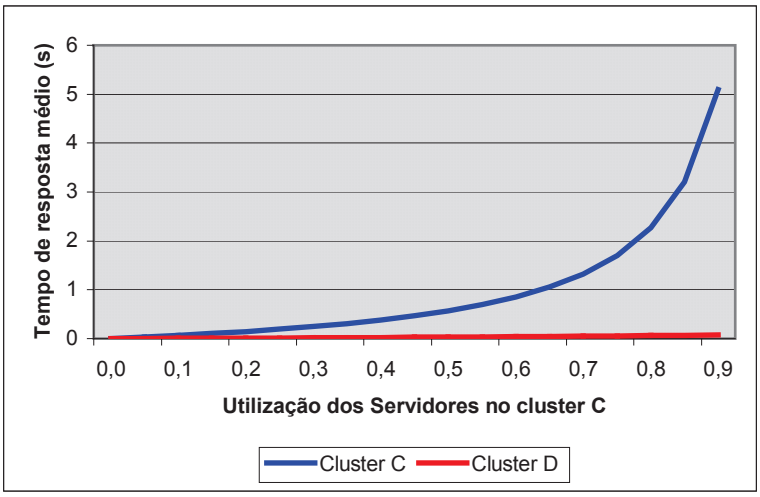

Figura 20 – Tempo médio de resposta em função da utilização no cluster C

A Tabela XVII mostra os valores das variáveis para o caso do custo função raiz quadrada. Para este caso, o cluster D obteve melhores resultados tanto no tempo de resposta quanto na utilização. Por este motivo, o cluster D foi escolhido como melhor opção nesta métrica.

TABELA XVII
VARIÁVEIS PARA O CASO DO MESMO CUSTO - FUNÇÃO RAIZ QUADRADA

λ	k	C	m	n	T_C(s)	T_D (s)	U_C	U_D
2.400.00	5	15	8.945	20.000	0,852	0,042	60%	26,83%
1.200.00	5	15	8.945	20.000	0,247	0,018	30%	13,41%
3.200.00	5	15	8.945	20.000	0,265	0,063	80%	35,77%
2.400.00	2	15	14.14	20.000	0,852	0,210	60%	42,42%

A Figura 21 apresenta o custo para o caso da função quadrática. É possível perceber a semelhança com o gráfico 5, onde nas utilizações mais baixas, o cluster D apresenta um tempo de resposta melhor e que vai se afastando do tempo de resposta do cluster C à medida que a utilização dos servidores vai crescendo. Quando o cluster C recebe 720.000 requisições por segundo, ele está com sua utilização em apenas 18%. Já o cluster D está com seus 800 servidores 90% utilizados. Para que a utilização de D chegue a 100%, são necessárias 800.000 requisições, fazendo com que o cluster C fique com 20% de utilização.

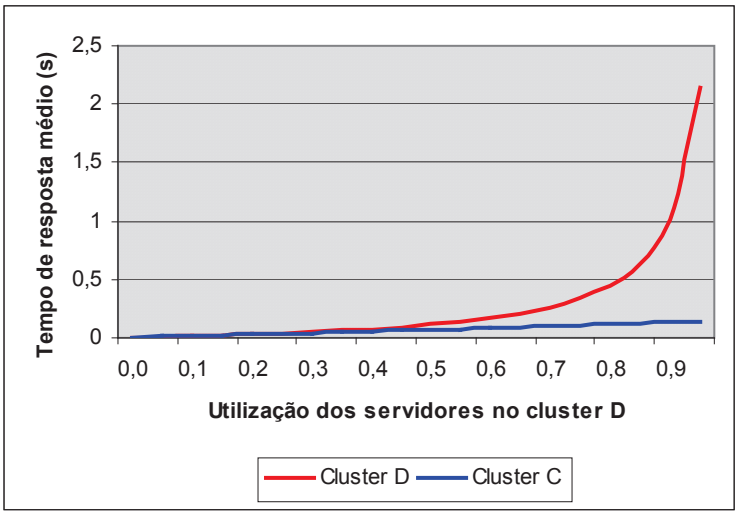

Figura 21 – Tempo médio de resposta em função da utilização no cluster D

A Tabela XVIII demonstra os valores obtidos para o caso do mesmo custo função quadrática. É possível observar que com o crescimento da utilização nos servidores, o cluster D vai aumentando seu tempo de resposta muito mais rapidamente do que o encontrado no cluster C. Neste caso, o cluster de melhor desempenho nesta métrica é o cluster C.

TABELA XVIII
VARIÁVEIS PARA O CASO DO MESMO CUSTO – FUNÇÃO QUADRÁTICA

λ	k	C	m	n	T_C(s)	T_D (s)	U_C	U_D
1.600.00	5	15	800	20.000	0,028	0,029	4%	20%
640.000	5	15	800	20.000	0,112	0,453	16%	80%
1.600.00	2	15	5.000	20.000	0,381	1,132	40%	80%
1.200.00	2	15	5.000	20.000	0,247	0,426	30%	60%

A Figura 22 mostra que o comportamento global encontrado para o caso da confiabilidade é o mesmo que pode ser verificado no exemplo da Figura 6. Usando a mesma utilização que foi apresentada para o cluster A para o cluster C (80%), observa-se agora que a utilização do cluster D é de 48%. O tempo de resposta de C é de 2,265 segundos, enquanto que o cluster D responde suas requisições com o tempo médio de 0,105 segundos. São necessários 20.000 máquinas no cluster C e 6.667 máquinas no cluster D para manter o quesito de confiabilidade como foi especificado anteriormente.

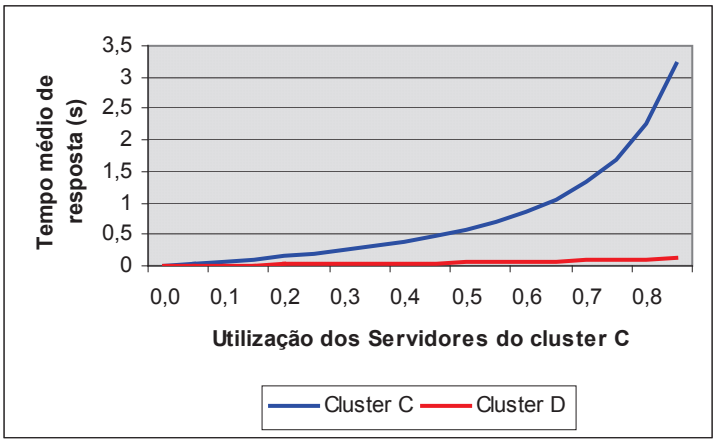

Figura 22 – Tempo de resposta médio em função da utilização no cluster C

Informações sobre a confiabilidade dos clusters podem ser observadas na Tabela XIX. Através destes números e da Figura acima, é possível observar que com o crescimento da carga o cluster D tem melhor comportamento mantendo seu tempo de resposta em um nível mais baixo do que o encontrado no cluster C. Por este motivo, o cluster D foi escolhido como melhor nesta métrica.

TABELA XIX
VARIÁVEIS PARA O CASO DA MESMA CONFIABILIDADE

λ	k	C	m	n	T_C(s)	T_D (s)	U_C	U_D
2.400.00	5	15	6.667	20.000	0,852	0,064	60%	36%
1.200.00	5	15	6.667	20.000	0,247	0,025	30%	18%
2.400.00	10	15	6.667	20.000	0,852	0,012	60%	18%
3.200.00	5	15	6.667	20.000	2,265	0,105	80%	48%

As Figuras 23 e 24 mostram a relação custo/desempenho para os clusters C e D, mostrando primeiro utilizando a função raiz quadrada e depois a função quadrática. O eixo y utiliza escala logarítmica. Para ambos os casos foi utilizado o valor de α igual a 100. É possível observar que o cluster C possui melhor relação custo/desempenho do que o cluster D no segundo caso, mas é possível observar o inverso no primeiro caso, onde o cluster D é melhor na relação custo/desempenho.

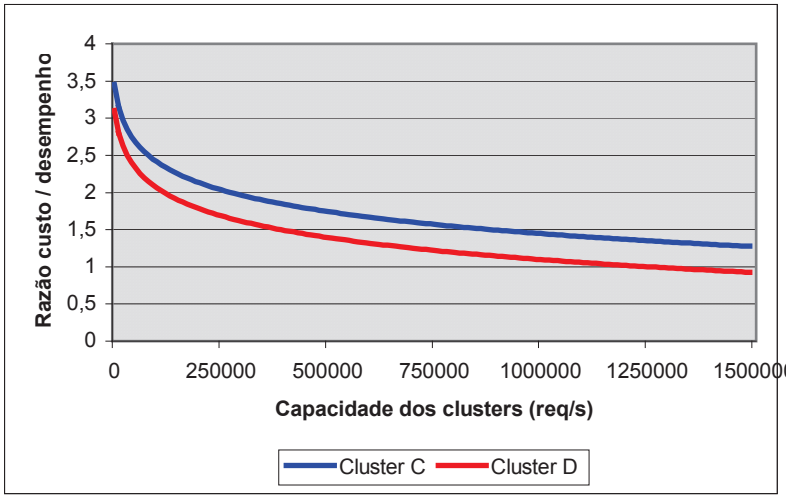

Figura 23 – Razão custo/desempenho em relação à capacidade dos clusters C e D utilizando a função custo raiz quadrada.

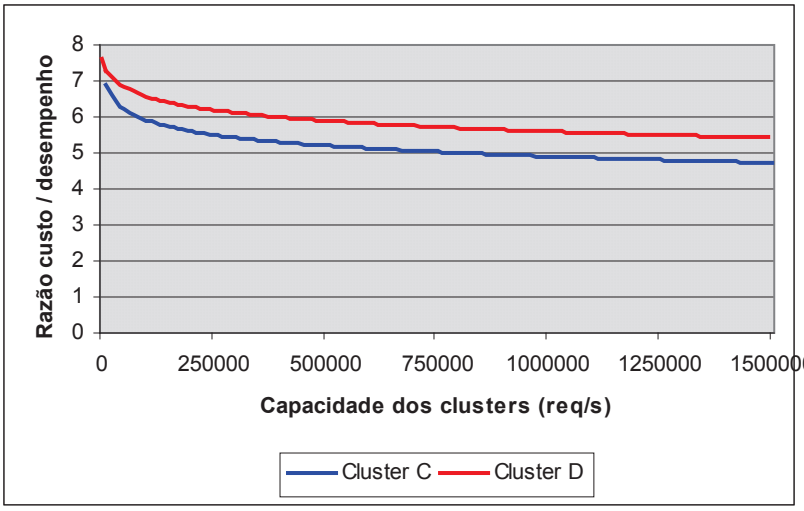

Figura 24 – Razão custo/desempenho em relação à capacidade dos clusters C e D utilizando a função custo quadrática.

A Tabela XX apresenta um sumário das comparações entre os clusters C e D, onde podemos verificar o tempo médio de resposta para o cluster D, o número de servidores no cluster D e a utilização dos servidores para os casos apresentados, considerando-se λ = 1.400.000 requisições por segundo. Desta vez a situação analisada foi com a utilização em 35% da capacidade do cluster C e um tempo de resposta médio de 0,3091 segundos.

Podemos acompanhar a variação de m, que tem variabilidade significativa. Para o caso do custo igual utilizando uma função quadrática, a utilização do cluster D novamente excede 100% e o tempo de resposta tende ao infinito.

TABELA XX
TAMANHO (M) REQUERIDO PARA O CLUSTER D PARA λ = 1.400.000 REQUISIÇÕES/SEGUNDO EM VÁRIOS CASOS

T_D (s)	m	U_D (%)	Caso
0,3092	6723	52	Igual tempo de resposta
0,0618	4000	35	Igual capacidade total
0,0219	8945	16	Igual custo: $C(x) = \sqrt{x}$
∞	800	> 100	Igual custo: $C(x) = x^2$
0,0310	6667	21	Igual confiabilidade do cluster

Na Tabela XXI é possível ver os resultados para λ = 2.400.000 requisições por segundo. Neste caso, a utilização do cluster C está em 60% e o tempo de resposta médio está em torno de 0,8525 segundos.

Para as duas taxas de chegada que foram analisadas, cuja síntese está apresentada respectivamente nas Tabelas XX e XXI, podemos observar grande variação no número de servidores do cluster B, bem como na utilização e no tempo de resposta. Devemos observar que alguns tempos de resposta podem ser inaceitáveis em determinadas situações, e este aspecto pode definir a escolha do cluster.

TABELA XXI
TAMANHO (M) REQUERIDO PARA O CLUSTER D PARA λ = 2.400.000 REQUISIÇÕES/SEGUNDO EM VÁRIOS CASOS

T_D (s)	m	U_D (%)	Caso
0,8525	7994	75	Igual tempo de resposta
0,1705	4000	60	Igual capacidade total
0,0424	8945	27	Igual custo: $C(x) = \sqrt{x}$
∞	800	> 100	Igual custo: $C(x) = x^2$
0,0645	6667	36	Igual confiabilidade do cluster

No caso mostrado na tabela acima é possível perceber a grande variabilidade no número de servidores de acordo com o caso escolhido. Como visto no capítulo anterior, nenhum dos clusters foi o melhor em todos os quesitos de comparação. Sendo assim, para escolher qual a composição do cluster utilizar, primeiro é necessário escolher qual a métrica mais importante para o usuário do cluster.

4.2 Comparando clusters de pequeno porte e alto poder de processamento

Nesta seção será mostrada a comparação de dois clusters com poucas máquinas, mas com poder de processamento individual maior. As variáveis serão configuradas da seguinte forma: n = 20, C = 15 e X = 500, para o cluster E. O cluster F será composto por máquinas que terão o poder de processamento três vezes maior (k = 3) em relação às máquinas do cluster E. Se algum valor diferente for utilizado, o mesmo será informado.

A Figura 25 mostra o comportamento da variável m em função de λ para os valores iguais a 5, 10 e 20. As curvas respeitam as equações 4 e 5 citadas no capítulo 3. Para o valor de λ igual a 6.000 requisições por segundo e n igual a 20, o cluster E fica 60% utilizado e o cluster F fica 80% utilizado. São necessários 5 servidores em F para igualar o tempo de resposta em 0,302 segundos.

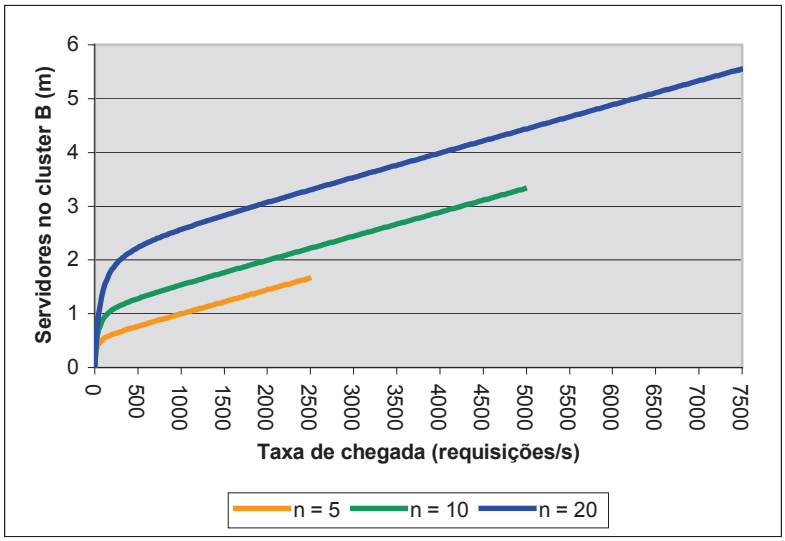

Figura 25 – Número de servidores no cluster F para o caso do mesmo tempo de resposta

A Tabela XXII demonstra alguns casos de teste para os clusters E e F. Em todos os casos o cluster F possui um maior valor de utilização para o mesmo tempo de resposta. Sendo assim, a melhor escolha neste caso é pelo cluster E.

TABELA XXII
VARIÁVEIS PARA O CASO DO MESMO TEMPO DE RESPOSTA

λ	k	C	m	n	T_E=T_F (s)	U_E	U_F
1.500	5	15	1	20	0,068	15%	60%
4.000	5	15	2	20	0,181	40%	80%
500	5	15	1	20	0,011	5%	20%
4.000	10	15	1	20	0,090	40%	80%

Na Figura 26 é possível ver o gráfico para o caso em que a utilização é igual em ambos os clusters. O tempo de resposta em F será sempre três vezes menor do que o observado no cluster E devido o valor da variável k utilizado neste exemplo.

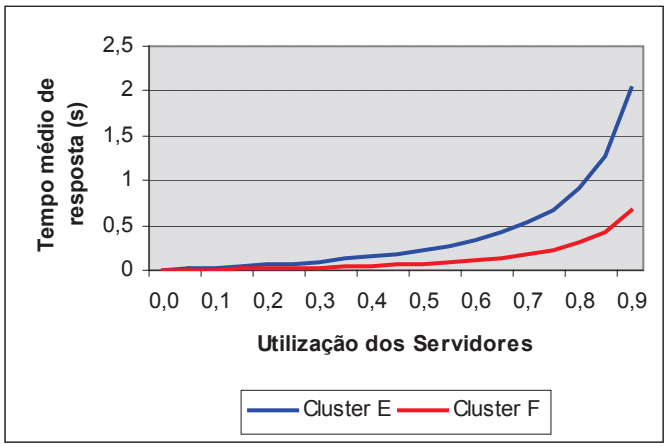

Figura 26 – Tempo médio de resposta em relação a utilização dos servidores para capacidades equivalentes

A Tabela XXIII demonstra as variáveis para o caso da mesma capacidade. O cluster F possui menor tempo de resposta com a mesma utilização. Sendo assim, para esta métrica o cluster F é a melhor escolha.

TABELA XXIII
VARIÁVEIS PARA O CASO DA MESMA CAPACIDADE

λ	k	C	X	m	n	T_E(s)	T_F (s)	U_E = U_F
6.000	5	15	500	4	20	0,341	0,068	60%
8.000	5	15	500	4	20	0,906	0,181	80%
6.000	10	15	500	2	20	0,341	0,034	60%
6.000	2	15	500	10	20	0,341	0,170	60%

A utilização em relação à comparação de mesmo custo utilizando a função raiz quadrada pode ser visualizada na Figura 27. Quando a utilização do cluster E ultrapassa 15%, o tempo de resposta começa a subir muito mais rápido do que no cluster F. Quando a utilização em E está em 60%, o cluster F está 33,3% utilizado e com tempo de resposta em 0,0406 segundos. O tempo de resposta em E é de 0,341 segundos, ou seja, aproximadamente 8,9 vezes maior. O número de servidores no cluster F é de 12 máquinas.

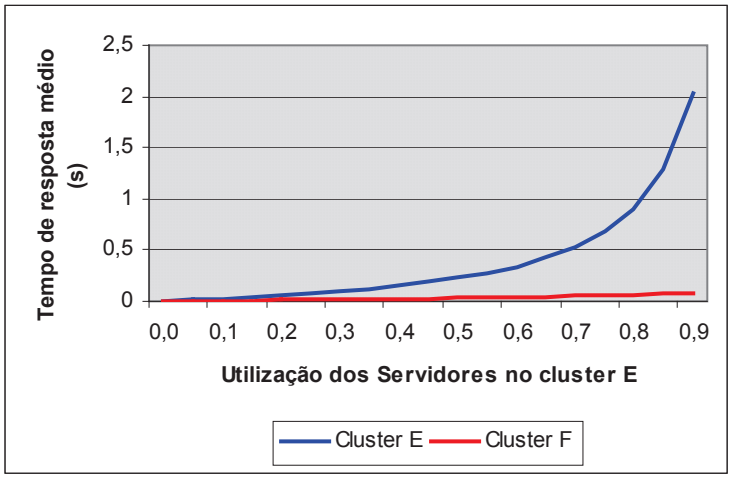

Figura 27 – Tempo de resposta médio em função da utilização no cluster E

A Tabela XXIV possui valores de teste para o caso do custo função raiz quadrada para os clusters E e F. Para todos os casos apresentados, o cluster F possui melhor comportamento observando a utilização e o tempo de resposta, sendo escolhido como melhor implementação para esta métrica.

TABELA XXIV
VARIÁVEIS PARA O CASO DO MESMO CUSTO – FUNÇÃO RAIZ QUADRADA

λ	k	C	m	n	T_E(s)	T_F (s)	U_E	U_F
6.000	5	15	9	20	0,341	0,016	60%	26,66%
8.000	5	15	9	20	0,906	0,025	80%	35,55%
2.000	5	15	9	20	0,585	0,004	20%	8,88%
6.000	10	15	7	20	0,341	0,005	60%	17,14%
6.000	3	15	12	20	0,341	0,040	60%	33,33%

Comparando o custo baseado na função de custo quadrática, podemos ver que o tempo de resposta fica muito próximo até quando o cluster F está 50%

utilizado. Acima deste valor, o tempo de resposta para o cluster F tem seus valores em crescimento acelerado. Para a utilização no cluster F em 80%, o cluster E fica 36% utilizado. Os tempos de resposta encontrados são 0,302 segundos para o cluster E e 0,0842 segundos para o cluster F. Neste caso, serão utilizados 3 servidores no cluster F. A Figura 28 demonstra este comportamento.

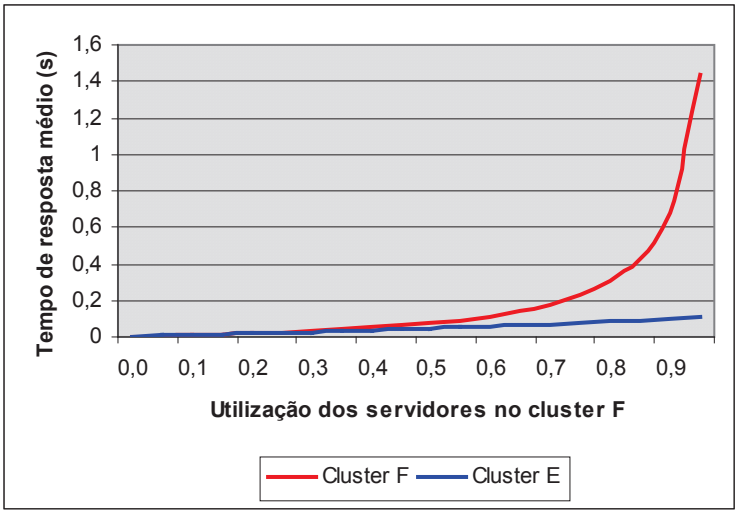

Figura 28 – Tempo de resposta médio em função da utilização no cluster F

A Tabela XXV mostra valores para o caso do mesmo custo função quadrática. Utilizando os dados da Tabela e da Figura acima, percebe-se que para níveis de utilização baixos, o cluster F tem o tempo de resposta melhor, mas mesmo assim bem próximo dos encontrados no cluster E. Com o aumento da carga, o cluster E passa a ter o melhor tempo de resposta, pois com a utilização mais baixa, o overhead gerado pelo tratamento das requisições é menor. O cluster que melhor se adapta a esta métrica é o cluster E.

TABELA XXV
VARIÁVEIS PARA O CASO DO MESMO CUSTO - FUNÇÃO QUADRÁTICA

λ	k	C	m	n	T_E(s)	T_F (s)	U_E	U_F
1.500	5	15	1	20	0,032	0,068	15	60
2.000	5	15	1	20	0,045	0,181	20	80
250	5	15	1	20	0,006	0,005	2,5	10
3.000	10	15	1	20	0,164	0,034	30	60

Na Figura 29 é mostrado o caso da mesma confiabilidade. Para o mesmo valor de confiabilidade utilizado nos casos anteriores, de 0,9 para cada servidor do cluster E e de 0,999 para os servidores do cluster F, o tempo de resposta no cluster F é sempre menor do que o encontrado no cluster E. Quando o cluster E está 70% utilizado, o tempo de resposta encontrado é de 0,529 segundos. Para igualar a confiabilidade, são necessários 7 servidores no cluster F de acordo com a equação 9, que fica 66,6% utilizado e respondendo as requisições em 0,1513 segundos em média.

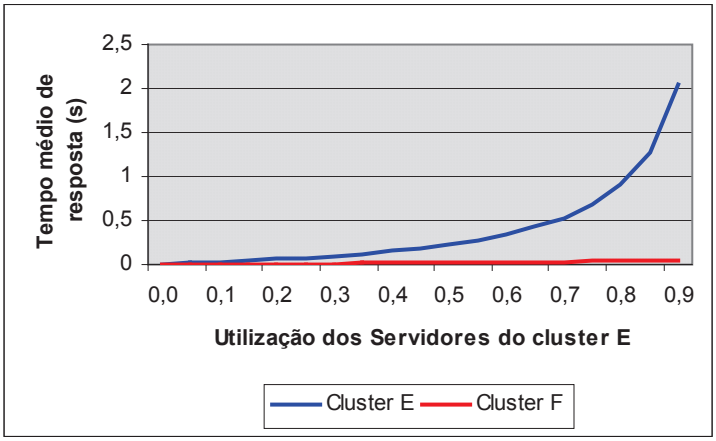

Figura 29 – Tempo de resposta médio em função da utilização no cluster E

O cluster F demonstra melhor comportamento nesta métrica de acordo com a Tabela XXVI para os valores de utilização e tempo de resposta. Sendo assim, a escolha para esta métrica é o cluster F.

TABELA XXVI
VARIÁVEIS PARA O CASO DA MESMA CONFIABILIDADE

λ	k	C	m	n	T_E(s)	T_F (s)	U_E	U_F
6.000	5	15	7	20	0,341	0,023	60%	34,28%
8.000	5	15	7	20	0,906	0,038	80%	45,71%
2.000	5	15	7	20	0,058	0,006	20%	11,42%
6.000	10	15	7	20	0,341	0,004	60%	17,14%

As Figuras 30 e 31 mostram a razão custo/desempenho para os clusters E e F. A Figura 30 mostra a comparação para o caso do custo função raiz quadrada e a Figura 31 para o caso do custo função quadrática. Ambas utilizam escala logarítmica no eixo y. A Figura 30 mostra que o custo/desempenho no cluster

F é melhor do que o encontrado no cluster E. O contrário pode ser verificado na Figura 31, onde a relação de custo/desempenho é melhor para o cluster E. As Figuras demonstram também que quanto maior a capacidade utilizada, mais os clusters se igualam quanto a essa métrica.

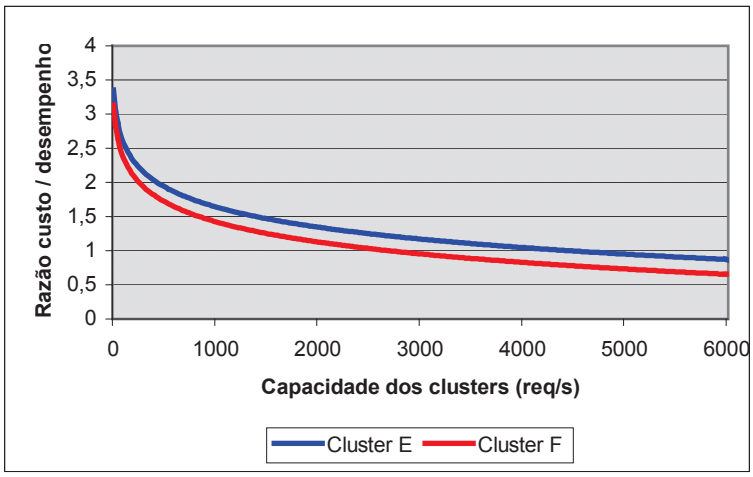

Figura 30 – Razão custo/desempenho em relação à capacidade dos clusters E e F utilizando o custo função raiz quadrada

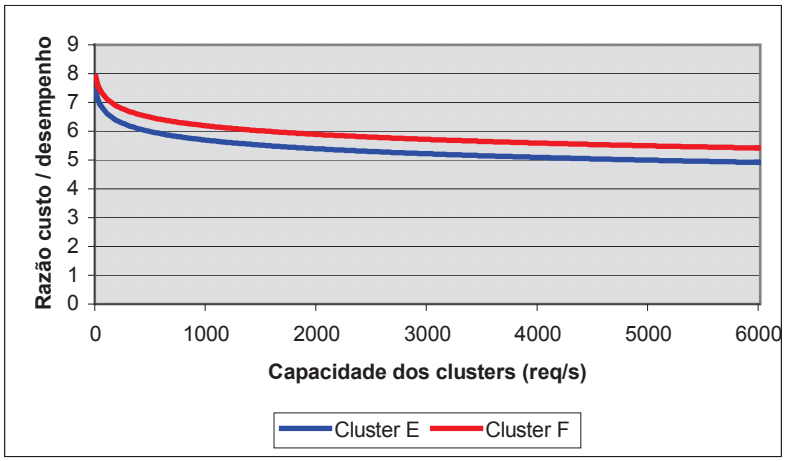

Figura 31 – Razão custo/desempenho em relação à capacidade dos clusters E e F utilizando o custo função raiz quadrada

A Tabela XXVII mostra o resumo das comparações quando o cluster E está 60% utilizado e possui 20 servidores. O número de requisições executadas pelos servidores é de 6.000 requisições por segundo. Para o caso do custo função quadrática, a utilização dos servidores do cluster F ultrapassa 100%. Os valores para os demais casos podem ser verificados na Tabela abaixo.

TABELA XXVII
TAMANHO (M) REQUERIDO PARA O CLUSTER F PARA λ = 6.000
REQUISIÇÕES/SEGUNDO EM VÁRIOS CASOS

T_F (s)	m	U_F (%)	Caso
0,0302	5	80	Igual tempo de resposta
0,1137	7	57	Igual capacidade total
0,0406	12	33	Igual custo: C(x) = $\sqrt{x}$
∞	3	> 100	Igual custo: C(x) = x^2
0,1011	7	57	Igual confiabilidade do cluster

A tabela apresenta grande variabilidade nos valores obtidos. Neste caso, assim como no caso apresentado no capítulo anterior, não foram possíveis encontrar os valores de m e n que fossem exatamente iguais aos obtidos nas fórmulas de cada caso. Por este motivo, pode-se perceber diferenças na tabela, como visto no caso da mesma capacidade, onde a capacidade do cluster A está em 60% e a observada para o cluster B é de 57%. Dependendo do caso escolhido, um dos clusters teve melhor desempenho do que o outro. Portanto, é importante definir qual métrica utilizar antes de escolher qual a composição do cluster.

CAPÍTULO 5
CONCLUSÃO

A implementação de um cluster Web depende de parâmetros interdependentes, onde uma forte relação entre eles existe e precisa ser bem estruturada. Estes parâmetros requerem a análise de parâmetros de desempenho, confiabilidade e custo para que os objetivos que levaram a formação do cluster sejam alcançados. Sendo assim, esta dissertação analisa estas relações e mostra como uma métrica influencia nas demais.

O modelo apresentado neste trabalho como base para a comparação é um modelo genérico para redes de filas abertas. Certamente o modelo é interessante e deve ser empregado para a análise preliminar e o planejamento da composição do cluster. No entanto, algumas simplificações foram feitas em benefício da generalização do modelo. Estas simplificações são discutidas a seguir.

Uma das vantagens básicas de um cluster é a possibilidade de agrupar um conjunto diverso de máquinas, com modelos e capacidades diferentes, para operar coletivamente como um único recurso. Assim, o conjunto de máquinas será heterogêneo, e não homogêneo, conforme assumido no modelo. No entanto, o modelo pode ser estendido para avaliar esta situação.

Outra limitação importante é a consideração de que o processo de chegada segue o modelo de Poisson. Vários trabalhos já demonstraram que os processos de chegada na Web e na Internet apresentam características de auto-similaridade e são mais complexos. No entanto, considerar estes novos modelos para o processo de chegada pode tornar inviável uma comparação como a apresentada nesta dissertação, uma vez que a solução destes novos modelos é muito mais complexa, quando ela existe.

Finalmente, características específicas da aplicação a ser executada no cluster devem ser consideradas na escolha do sistema. Por exemplo, a busca realizada pelo Google é passível de grande paralelização, a carga é muito intensa e o objetivo do sistema é ter a melhor relação custo/desempenho. Este tipo de composição de perfil de carga e objetivo do sistema ajuda a direcionar a escolha e

deve ser investigado para cada aplicação específica. Há vários tipos de servidores Web, servidores de email, servidores de comércio eletrônico, servidores de informação, e outros.

Observamos que a composição dos clusters e o desempenho obtido podem variar muito quando cada critério é equiparado. Esta variação torna imprescindível este tipo de análise. Há várias possibilidades para trabalhos futuros. Um dos problemas dos clusters é a energia consumida pelo sistema, para sistemas de grande porte. Uma extensão deste modelo pode considerar também o consumo de energia ou o espaço ocupado pelo cluster (principalmente os de grande porte, como o do Google) como outros parâmetros de pesquisa.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] Arnold O. Allen, “Probability, Statistics, and Queueing Theory with Computer Science Applications”, Second edition, Academic Press, 1990.

[2] J. Almeida, V. Almeida, D. J. Yates. “Measuring the Behavior of a World Wide Web Server”, Seventh IFIP Conference on High Performance Networking, White Plains, NY, April 1997.

[3] M. F. Arlitt, C. L. Williamson. “Internet Web Servers: Workload Characterization and Performance Implications”. IEEE/ACM Transactions on Networking, 5(5):631-645, 1997.

[4] M. Arlitt, C. Williamson, “Web Server Workload Characterization: the Search for Invariant”, In Proceedings 1996 ACM SIGMETRICS Conference Measurement Computer Systems, Philadelphia, Pennsylvania, maio de 1996.

[5] L. A. Barroso, J. Dean, H. Holzle, “Web Search for a Planet: The Google Cluster Architecture”, IEEE Micro, pp. 22-28, March-April 2003.

[6] V. Cardellini, E. Casalicchio, “The State of the Art in Locally Distributed Web-Server Systems”, ACM Computing Surveys, 34(2), pp. 263-311, June 2002.

[7] M. E. Crovella, Azer Bestavros. “Self-Similarity in World Wide Web traffic: Evidence and possible causes”. IEEE/ACM Transactions on Networking, 5(6):835-846, December 1997.

[8] M. E. Crovella, R. Frangioso, M. Harchol-Balter. “Connection Scheduling in Web Servers”, In USENIX Symposium on Internet Technologies and Systems, Boulder, Colorado, October 99, pp. 243-254.

[9] M. E. Crovella, M. S. Taqqu, A. Bestavros. “Heavy-tailed probability distributions in the World Wide Web”. In A Practical Guide to Heavy Tails, pages 3-26. Chapman & Hall, New York, 1998.

[10] J. Dilley, R. Friedrich, T. Jin, J. Rolia, “Measurement Tools and Modeling Techniques for Evaluating Web Server Performance”. Computer Performance Evaluation, 1245:155-168, junho de 1997.

[11] M. Harchol-Balter, N. Bansal, B. Schroeder. “Implementation of SRPT Scheduling in Web Servers”, School of Computer Science, Carnegie Mellon University, November 2000.

[12] Y. Hu, A. Nanda, Q. Yang, “Measurement, Analysis and Performance Improvement of the Apache Web Server”. 18th IEEE International Performance Computing and Communications Conference, fevereiro de 1999.

[13] R. Jain, “The Art of Computer Systems Performance Analysis”, John Wiley& Sons, 1991.

[14] Keynote Systems. Disponível em www.keynote.com, acessado em agosto de 2004.

[15] L. Kleinrock, “Queueing Systems: Volume I: Theory”, John Wiley & Sons, New York, 1975.

[16] K. R. T. Larsen, P. A. Bloniarz, “A Cost and Performance Model for Web Service Investment”, Comm. ACM, 43(2): 109-116, fevereiro de 2000.

[17] E. D. Lazowska, J. Zahorjan, G.S. Graham, K. C. Sevcik, “Quantitative System Performance – Computer System Analysis Using Queueing Network Models”, Prentice-Hall, 1984.

[18] R. E. McGrath, “Performance of Several Web Server Platforms”, Disponível em http://www.ncsa.uiuc.edu/InformationServers/Performance/Platforms/report.html, acessado em julho de 2004.

[19] D. A. Menascé, V. A. Almeida, “Planejamento de Capacidade para Serviços na Web – Métricas, Modelos e Métodos”. Editora Campus, Rio de Janeiro, 2002.

[20] D.A. Menascé, “Trade-offs in Designing Web Clusters”, IEEE Internet Computing, pp 1-4, Sep-Oct 2002.

[21] Microsoft Brasil. Disponível em www.microsoft.com.br, acessado em outubro de 2004.

[22] Cristina D. Murta, Géri N. Dutra, “Modeling HTTP Service Times”, Proceedings of the IEEE GLOBECOM 2004, Dallas, Texas, November 2004.

[23] NCSA Httpd. Disponível em http://hoohoo.ncsa.uiuc.edu/, acessado em outubro de 2004.

[24] NetCraft. Disponível em www.netcraft.com, acessado em outubro de 2004.

[25] D. P. Olshefski, J. Nieh, D. Agrawal. “Inferring Client Response Time at the Web Server”, in Proceedings of the ACM Sigmetrics, 2002.

[26] Kihong Pak, Walter Willinger (Editors), “Self-Similar Network Traffic and Performance Evaluation”, John Wiley & Sons, 2000.

[27] D. Patterson, J. Hennessy, “Computer Architecture: A Quantitative Approach”, Third edition, Morgan-Kauffman Publishers, 2000.

[28] Projeto Apache. Disponível em www.apache.com, acessado em setembro de 2004.

[29] P. Rodriguez, K. W. Ross, E. W. Biersack. “Improving the WWW: Caching or Multicast?”, Computer Networks and ISDN Systems, Vol. 30, November 1998.

[30] A. O. Salam-Alada, A. Waheed.“Performance Comparison of Apache and Microsoft IIS Web Servers”, In Proceedings of ACIT - International Arab Conference on Information Technology 2002, Doha, Qatar, December 2002.

[31] L. F. G. SOARES, "Redes de Computadores - Das LAN’s, MAN’s e WAN’s às Redes ATM". Editora Campus, 1995.

[32] Specweb. Disponível em www.spec.org, acessado em janeiro de 2005.

[33] K. Suryanarayanan, K. J. Christensen. “Performance Evaluation of New Methods of Automatic Redirection for Load Balancing of Apache Servers Distributed in the Internet”, Proceedings of the IEEE Conference on Local Computer Networks, pp. 644-651, November 2000.

[34] TPC-W. Disponível em www.tcp.org, acessado em janeiro de 2005.

[35] K. S. Triverdi, “Probability & Statistics with Reliability, Queuing and Computer Science Applications”, Prentice Hall, Upper Saddle River, 1982.

[36] Alex Vrenios, “Linux Cluster Architecture”, Sams Publishing, Indianapolis, 2002.

[37] M. Welsh, D. Culler. “Adaptive Overload Control for Busy Internet Servers”, In Proceedings of the 4th USENIX Conference on Internet Technologies and Systems (USITS'03), March 2003.

[38] H. Xie, L. Bhuyan, Y. Chang. “Benchmarking Web Server Architectures: A Simulation Study on Micro Performance”, Fifth Workshop on Computer Architecture Evaluation using Commercial Workloads, Cambridge, Massachusetts, February 2002.