Arquitetura do Google Cluster

Anísio M. Lacerda[1], Frederico P. Quintão[1], Vanessa C. Sabino[1]
[1] Programa de Residência em Tecnologia
UFMG – Google

Resumo—

Este artigo apresenta alguns pontos publicamente conhecidos da arquitetura de cluster do Google. Para entender os principais benefícios trazidos por essa arquitetura, será explicada a aplicação mais importante a que ela atende, a máquina de buscas Google, que possui características bastante distintas da maioria das aplicações encontradas em *data centers* tradicionais. Veremos que, devido a grande capacidade de paralelização no atendimento às requisições de busca, torna-se viável a criação dos clusters utilizando um grande número de PCs comuns, ao invés de um menor número de servidores de maior performance, o que permite atingir uma melhor razão preço/performance. Apesar do sigilo quanto a informações mais precisas sobre o hardware utilizado no Google, alguns artigos publicados por funcionários da empresa fornecem a base para ilustrar os princípios seguidos na definição da arquitetura, além de indicar suas principais tendências. Em particular, abordaremos a questão da energia, que tem crescido em importância nos últimos anos, tornando-se o foco de artigos mais recentes.

Keywords— **Arquitetura de Computadores, Cluster, Tendências.**

I. Introdução

Em apenas 10 anos, o Google deixou de ser uma idéia de alunos do Departamento de Ciência da Computação da Stanford University para se tornar a maior empresa de Internet do mundo, com ações ultrapassando a casa dos 150 bilhões de dólares, centros de pesquisa espalhados pelo mundo, 5000 engenheiros dedicados à pesquisa em Tecnologia da Informação e a impressão de que não se trata de uma nova bolha, mas sim um novo império que surge na área de computação.

Todo esse sucesso se deve, obviamente, à competência de seus criadores (L. Page e S. Brin) e seguidores, mas também devido à qualidade do trabalho e inovação inerentes ao espírito Google. Essas duas características – qualidade e inovação – podem ser admiradas em todos os serviços que a empresa disponibiliza na Internet – interfaces bem construídas, minimalistas, informação requisitada sempre em mãos, ambientes personalizados. Entretanto, toda essa parafernalha de software e dados só se encontra disponível graças a uma arquitetura de hardware muito bem construída. Por baixo das máquinas de busca Google, do Orkut, do Google Maps, Picasa e tantos outros serviços, está um conjunto de computadores espalhados pelo globo, que permitem atender aos usuários do sistema com tamanha qualidade. A esse conjunto de computadores dá-se o nome do Google Cluster – trata-se de um dos maiores *data centers* já construídos em todo o mundo, com milhares de computadores. A primeira geração do Google Cluster tinha cerca de 15 mil computadores. Estima-se que hoje o Google Cluster teria, no mundo todo, entre 50 e 70 mil computadores.

O objetivo do presente artigo é apresentar o requisito principal que estimulou a criação de toda essa infra-estrutura de hardware – a Máquina de Buscas Google. O artigo também apresenta a solução adotada pelos engenheiros do Google para conseguir atender os requisitos dessa aplicação. Como será mostrado, um dos detalhes mais interessantes da arquitetura proposta é que ela se baseia em PCs comuns, ao invés de utilizar servidores de ponta. Finalmente, o artigo termina com uma discussão sobre as principais tendências que aparentemente estão na pauta dos engenheiros do Google: *chip multiprocessor* e provisionamento de energia, principalmente.

II. Processamento de uma consulta

Os servidores do Google atendem, em momentos de pico, a requisições de milhares de buscas por segundo, em que cada uma lê centenas de megabytes de dados e consome bilhões de ciclos de CPU [1].

Os passos básicos da requisição de uma busca são os seguintes: Primeiramente o usuário entra no site do Google através de sua URL, o que faz com que o navegador acesse um servidor de DNS para procurar um endereço IP correspondente. Há vários clusters do Google preparados para atender essas requisições, distribuídos em diversas partes do mundo, de forma a prover capacidade suficiente para o tráfego de dados e evitar interrupções do serviço que poderiam ser causadas devido a problemas em algum *data center*. Um sistema de balanceamento de carga baseado em DNS seleciona o cluster de forma a minimizar o tempo de resposta de acordo com a posição geográfica do usuário e a capacidade disponível dos vários clusters. Então o navegador envia a requisição contendo os termos da busca para o cluster selecionado, que irá processá-la localmente, sempre utilizando balanceamento de carga entre seus diversos servidores.

A máquina que recebe a requisição (chamada de GWS – Google Web Server) coordena sua execução e formata o resultado na página que será retornada ao usuário. A execução da requisição de busca inicia-se com uma consulta dos termos em arquivos invertidos [7], que mapeiam cada termo a uma lista de documentos nos quais ele ocorre, e então é feita uma intersecção dessas listas e o resultado é ordenado por relevância de acordo com o algoritmo do *PageRank* [5], entre

outras métricas. Apesar do índice completo conter terabytes de dados, a consulta é feita paralelamente em diversas partições menores do índice, chamadas *index shards*, cada uma contendo apenas um subconjunto de documentos. Cada um dos *index shards* está replicado em várias máquinas, de forma a garantir a confiabilidade e desempenho. Após obter os identificadores dos documentos e ordená-los por relevância, os servidores de documento são utilizados para obter as demais informações que serão incluídas no resultado, tais como título, URL e trecho do documento onde aparecem os termos. As mesmas estratégias dos servidores de índice são utilizadas aqui: distribuição aleatória dos documentos em partições menores, replicação de cada uma dessas partições em diversos servidores e balanceamento de carga para rotear as requisições. Outros processamentos que acontecem antes de retornar os resultados para o usuário incluem verificação ortográfica e geração de anúncios relevantes.

A figura a seguir esquematiza todos os passos envolvidos no atendimento de uma requisição de busca:

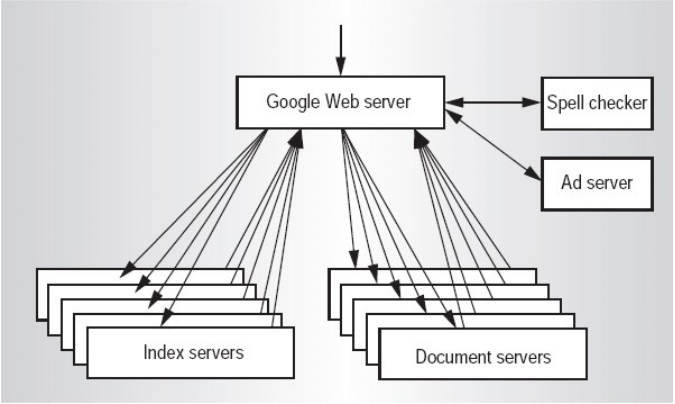

Figura 1. Processamento de uma consulta – Fonte: [5]

III. Arquitetura de Cluster Google

A. Principais fatores

Alguns fatores orientam as decisões tomadas para a construção da arquitetura de cluster do Google. Os principais são eficiência de energia e razão preço/performance. O consumo de energia tem se tornado um gargalo nas operações do Google, como veremos adiante, até mesmo devido a limites físicos dos *data centers*. E a razão preço/performance é um fator cuja utilização é viável para as aplicações do Google pois a performance máxima de cada processador é algo menos importante, devido á facilidade de paralelização das aplicações, o que as torna orientadas a *throughput*. A capacidade de paralelização é decorrente tanto do fato de que requisições diferentes podem rodar em processadores diferentes como também uma única requisição pode rodar em múltiplos processadores, devido ao particionamento dos índices, servidores de documento, etc.

B. Solução

Os dados a seguir foram obtidos de um artigo publicado em 2003 [1], servindo para ilustrar as decisões de arquitetura tomadas pelo Google, mas não refletindo com exatidão a situação atual de seus clusters.

A infraestrutura de computação dos clusters do Google é criada a partir de mais de 15 mil PCs comuns, de baixa confiabiliade, porém utilizando software para controle de tolerância a falhas, que ficam responsáveis por replicar os serviços em várias máquinas e detectar e tratar as falhas automaticamente. Dessa forma, economiza-se dinheiro que de outra forma seria gasto em servidores de alta performance e itens como fontes reduntantes, RAIDs e componentes de maior qualidade.

Além disso, o design é feito de forma a maximizar *throughput* ao invés de tempo de resposta de servidor, já que o tempo de resposta total é gerenciado através da paralelização das requisições. Assim, ao escolher uma máquina, a performance por *thread* é um fator de menor importância se a máquina for capaz de realizar bastante trabalho por unidade de tempo.

De forma geral, os acessos ao índice e outras estruturas de dados são realizados de forma *read-only*, o que permite eliminar vários dos problemas que geram *overhead* para garantir consistência que surgem, por exemplo, em aplicações baseadas em bancos de dados comuns. Para atualização dos dados, que são bem mais raras, as requisições são desviadas dos servidores envolvidos na replicação.

Também é explorado agressivamente o paralelismo da aplicação, particionando tudo que é possível e depois realizando o *merge*, que não é tão caro. No caso dos servidores de índice, por exemplo, adicionar novas máquinas em um *pool* responsável por algum *index shard* aumenta a capacidade, enquanto novos *shards* acomodam o crescimento do índice. Conforme explicado por Barroso, "Através da paralelização da busca sobre diversas máquinas, nós reduzimos a latência média necessária para responder uma *query*, dividindo a computação total em mais CPUs e discos" [1]. Como não é necessário comunicação entre os diversos *index shards*, o aumento de performance ao adicionar novas máquinas é quase linear. É interessante notar que, ao mesmo tempo que a adição de novas máquinas que replicam os serviços é importante para obter capacidade suficiente, isso também serve para aumentar a tolerância a falhas.

Dessa forma, no Google não se aplicam os requisitos que justificariam o uso de servidores mais caros, tais como uma razão de computação/comunicação baixa, padrões de comunicação ou particionamento dinâmicos ou difíceis de

prever. Então, o uso de PCs comuns reduz o custo da computação, o que possibilita o uso de mais recursos computacionais por requisição, sendo possível empregar técnicas de ranking mais complexas ou buscar em índices maiores de documentos.

B.1 As Máquinas

Em 2003, os *racks* do Google eram compostos por 40 a 80 servidores que variavam entre Celerons de um processador de 533Mhz até Pentiums III dual de 1.4Ghz, cada um contendo um ou mais drivers IDE com capacidade de 80Gb. As máquinas de cada lado do rack se conectam através de um *switch* Ethernet de 100-Mbps que tem um ou dois gigabit *uplinks* para um gigabit *switch* central que conecta todos os *racks*.

De forma geral, os servidores de índice têm menos espaço em disco, pois têm uma carga mais voltada a processamento, enquando os servidores de documento possuem mais capacidade de armazenamento. Os servidores do Google em geral são trocados a cada dois ou três anos, devido a disparidades de performance que causam dificuldades com o balanceamento de carga [4], portando atualmente estima-se que os servidores estejam duas ou três gerações mais atualizados.

No Google, o critério mais importante de seleção de hardware é o "custo por query, expresso como a soma do custo de capital (com depreciação) e custos de operação (hospedagem, administração de sistema e reparos) dividido pela performance" [1]. Uma comparação da performance e custo entre servidores de ponta e computadores x86 típicos mostra sua vantagem:

Por uma cotação de 2002, um rack com 88 dual-CPU 2Ghz Intel Xeon com 2Gbs de RAM e 80 Gbs de disco, totalizando 176 2Ghz Xeon CPUs, 176 Gbs de RAM e 7 Tbytes de espaço de disco, estava sendo oferecido a US$278.000. Ao mesmo tempo, um servidor baseado em x86 típico contém oito 2Ghz Xeon CPUs, 64 Gbs de RAM e 8 Tbytes de espaço de disco, custando US$758.000, ou seja, o servidor custa aproximadamente três vezes o valor da solução com computadores mais simples, e tem 22 vezes menos CPU, um terço da RAM e pouca diferença de espaço em disco. A diferença de custo costuma ser justificada pela maior banda de interconexão e confiabilidade do servidor, porém a arquitetura de alta redundância do Google torna isso desnecessário.

Apesar do aumento de custo em administração de sistema e reparos na solução adotada, a homogeneidade da aplicação facilita seu gerenciamento e o aumento do tamanho do cluster, por exemplo de centenas para milhares, não tem grande impacto.

O servidor de índice é o componente da infraestrutura que tem maior impacto na razão preço/performance e portanto será analisado em maior detalhe. Ele apresenta as seguintes características em nível de instrução, ao ser executado em um Pentium III dual de 1Ghz:

Ciclos por instrução	1,1
Branch mispredict	5,0%
Level 1 instruction miss*	0,4%
Level 1 data miss*	0,7%
Level 2 miss*	0,3%
Instruction TLB miss*	0,04%
Data TLB miss*	0,7%

Tabela I
DETALHES DA APLICAÇÃO GOOGLE – FONTE: [1]

* por instrução executada.

Na aplicação em questão, o pipeline fica exposto a latências da DRAM, pois as estruturas de dados são grandes demais para o cache do processador e há dependência de dados no controle do fluxo. Isso causa dificuldade para *branch prediction* e portanto não é possível explorar muito ILP, e assim não há grande aproveitamento das técnicas mais agressivas de execução presentes em processadores mais modernos. Em média é enviada uma nova instrução apenas a cada dois ciclos de clock, apesar das máquinas utilizadas suportarem até três instruções por clock.

Assim, é mais vantagem explorar o paralelismo intrínseco da aplicação através de um maior número de nós no cluster, e também paralelismo de threads no nível da microarquitetura, tais como *simultaneous multithreading* (SMT) e *chip multiprocessor* (CMP). Experiências realizadas com SMT atingiram o aumento de performance máximo especificado pelo fabricante, e no caso de CMP acredita-se que seria atingido um aumento de performance quase linear de acordo com o número de *cores*, havendo um cache L2 compartilhado de tamanho suficiente para acelerar a comunicação interprocessos.

Em relação à memória, observa-se na tabela acima boa performance do cache e buffer de instruções, o que é explicado por características particulares da aplicação.

Os blocos de dados do índice não possuem característica de localidade temporal, devido ao tamanho do índice e padrões de acesso difíceis de prever. Porém, dentro do bloco de dados, há o benefício da localidade espacial, que pode ser explorada por *hardware prefetching* e linhas de cache maiores, o que é suficiente para garantir um *miss* de cache reduzido.

Os dados obtidos por Barroso [1] indicam que é utilizado menos de 20% do bus de memória, e portando esse não é um gargalo. A explicação é a quantidade de computação média necessária para cada linha de cache dos dados do índice que são trazidos para o cache do processador e o fato

da requisição de mais dados ser dependente dos dados calculados. Portanto, o tamanho das linhas de cache é um fator importante na performance.

IV. Tendências

A capacidade instalada de energia nos *data centers*, seu custo associado e problemas de resfriamento têm sido assuntos recorrentes em artigos sobre a arquitetura do Google, o que demonstra grande preocupação da empresa nesse aspecto.

O TCO (*Total Cost of Ownership*) de um cluster de grande escala possui quatro componentes principais [2]:

$$TCO(Cluster) = \begin{array}{l}\text{preço do hardware} + \text{energia} \\ + \text{operação} + \text{software}\end{array}$$

O Google possui um perfil de TCO diferente da maioria das empresas, pois seu custo de software é reduzido devido ao fato dele produzir internamente tudo o que utiliza, alavancando o que está disponível pela comunidade *open-source*. Dessa forma, o foco torna-se o hardware e os custos relacionados a energia.

O gráfico abaixo, que mostra performance comparada a preço do servidor e consumo de energia em sucessivas gerações de hardware do Google, demonstra que performance por watt é um fator que não apresentou melhorias significativas no decorrer do tempo, aumentando sua importância relativa no TCO. Em outras palavras, cada ganho em performance tem sido acompanhado por um aumento proporcional em consumo de energia.

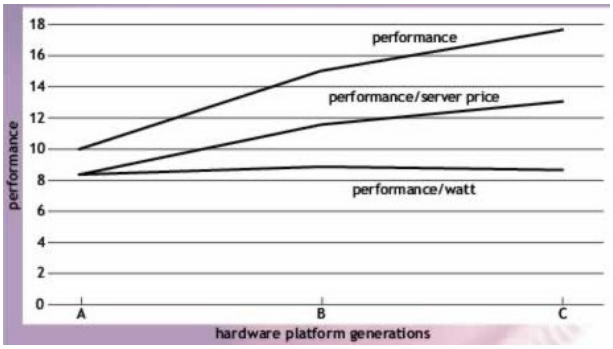

Figura 2. Performance de 3 gerações de servidores Google – Fonte: [2]

O cálculo da energia consumida por cada servidor no caso médio da arquitetura descrita anteriormente é de 120W, sendo 55W para a CPU, 10W para o HD, 25W para a RAM e considerando uma eficiência de 75% para uma fonte ATX. Para um *rack* desses processadores, temos um total de 10kW, o que resulta em 400W/ft^2. Para processadoes mais rápidos, a densidade ultrapassava 700W/ft^2. Isso excede muito a densidade de energia para *data centers* comerciais, que fica entre 70 e 150W/ft^2, sendo então necessário resfriamento especial ou espaço adicional para reduzir a densidade de energia a níveis toleráveis. O custo de construir tais *data centers* é significativo, ficando entre US$10 e US$20 por Watt utilizado em momentos de pico [3], excluindo o resfriamento e outras cargas auxiliares. Portanto, há um incentivo econômico para operar na capacidade máxima desses locais, de forma a amortizar melhor esse custo inicial.

Em dados de 2005 [2], temos que o servidor custaria em torno de US$3.000 e consumiria em média 200W, ultrapassando 300W em momentos de pico. Ineficiências na distribuição da energia e gastos com resfriamento dobrariam esses valores. Considerando um custo de nove centavos (de dólar) por kW/h e quatro anos de tempo de vida do servidor, temos que o custo de energia para esse servidor chega a 40% do custo do hardware.

E a situação pode ficar ainda pior: se a performance dos processadores continuar aumentando na mesma taxa, e a relação performance/watt continuar seguindo a tendência atual, em poucos anos o custo de energia de um *data center* irá superar – por uma larga margem – o preço do hardware. Nesse caso, especula Barroso [2], poderão aparecer modelos de negócio absurdos – nos quais a companhia de energia provê hardware gratuito para os clientes que assinam contratos de longo prazo. Além do impacto nos modelos de negócio e na sobrevivência dos grandes *data centers*, o consumo de energia também tem impacto na saúde geral do planeta.

Portanto, servidores de baixo consumo de energia são atrativos para esses clusters de grande escala, porém outros fatores também devem ser considerados: nada adianta se a redução de energia for seguida por redução proporcional de performance, e também os servidores não podem ser consideravelmente mais caros. É necessário considerar watts por unidade de performance e comparar custo de depreciação dos servidores com a economia de energia.

A. CMP – A solução para computação eficiente

A utilização de *chip multiprocessing* é uma das soluções adotadas para reduzir o consumo de energia. Nesse tipo de ambiente, técnicas especulativas precisam ser muito precisas para justificar a energia extra que demandam, já que há outras threads com instruções não especulativas prontas para serem executadas. Além disso, infelizmente a maior parte dos *workloads* para servidores apresentam pouca possibilidade para paralelização em nível de instrução (e portanto eles não são objeto da extrema computação especulativa existente em processadores modernos).

De acordo com Barroso [2], boa parte das aplicações do Google possuem esta característica. Há pouca possibilidade para explorar ILP: isto acontece devido às estruturas de da-

dos que são utilizadas e ao fluxo de dados, que possui pouca localidade de referência, dado que as consultas dos usuários possuem pouca localidade e são impossíveis de prever. Entretanto, este mesmo *workload* apresenta excelente *speed-up* em multi-processadores, CMPs, etc.

CMPs são eficientes em termos de energia devido ao progresso tecnológico alcançado nos últimos anos. Por exemplo, recentemente Intel e AMD introduziram projetos CMP com um *package* que é aproximadamente o mesmo usado nos single-cores. Por exemplo, o Opteron dual-core possui performance 1.8 vezes melhor do que a versão single-core, com um consumo de energia apenas 7% pior [6]. De fato, esta é a primeira vez na história em que há um investimento massivo para aumento da eficiência dos processadores em termos de energia.

Infelizmente, a venda de CMPs ainda apresenta alguns empecilhos, tais como:

- Marketing: Megahertz (MHz) é uma unidade de medida mais fácil de entender e de comunicar (e portanto, é mais fácil de vender).
- Threads não estão em todo lugar: ainda existem muitas aplicações que precisam ser migradas para ambientes de processamento paralelo. Além disso, existe um certo receio por aplicações paralelas, dado que elas são muito mais complexas de serem construídas e mantidas. Entretanto, conforme afirmado por Barroso [2], existem diversos avanços na área de compiladores e bibliotecas para facilitar a vida dos desenvolvedores.

Apesar das dificuldades acima frisadas, Barroso [2] estima que em breve um número maior de processadores CMPs existirá no mercado, bem como de aplicações, porque esta é talvez a única solução a curto prazo para solução do problema de consumo de energia em *data centers*.

V. Conclusão

O uso de PCs comuns, com sua informação replicada em várias máquinas, permite ao Google atender a sua carga massiva de requisições, com alta performance e confiabilidade, a um custo menor do que se fossem utilizados servidores de ponta. Isso é possível devido a paralelização da aplicação, que acontece tanto no ato de atender várias requisições independentes simultaneamente como também no processamento da própria requisição, cujos dados necessários estão espalhados por diversas máquinas que podem ser tratadas de forma independente durante a maior parte do processo.

Porém, enquanto a adição de novas máquinas é capaz de gerar um aumento de performance quase linear, um fator limitante da capacidade dos clusters e que incorre em um custo significativo é o consumo de energia. Esta é uma questão que ainda não está resolvida, mas algumas iniciativas tais como o uso de CMP e escalonamento de voltagem têm apresentado bons resultados.

Referências

[1] U. Hoelzle, J. Dean, and L. Barroso. Web Search for A Planet: The Architecture of the Google Cluster. In *IEEE Micro Magazine*, April 2003.

[2] L. Barroso. The Price of Performance: An Economic Case for Chip Multiprocessing. In *ACM Queue*, September 2005.

[3] X. Fan, W. Weber, and L. Barroso. Power Provisioning for a Warehouse-sized Computer. In *Proceedings of the 34th ACM International Symposium on Computer Architecture*, San Diego, CA, June 2007.

[4] C. Badue, R. Baeza-Yates, B. Ribeiro-Neto, A. Ziviani, and N. Ziviani. Analyzing Imbalance among Homogeneous Index Servers in a Web Search System. In *Information Processing Management*, Volume 43, Issue 3, 2007.

[5] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. In *Computer Networks and ISDN Systems*, 30:107–117, 1998.

[6] AMD competitive server benchmakrs; http://www.amd.com/us-en/Processors/ProductInformation/0,,30_118_8796_8800 97051,00.html

[7] Witten, Ian H. and Witten, Ian H. and Moffat, Alistair and Bell, Timothy C. *Managing Gigabytes: Compressing and Indexing Documents and Images*, Morgan Kaufmann 1999