Avaliação do Sistema de Arquivos Paralelo do Cluster Netuno

Juliana Correa, Gabriel P. Silva
DCC-IM/UFRJ
jucorrea@ufrj.br, gabriel@dcc.ufrj.br

Resumo

Este artigo apresenta uma avaliação do sistema de arquivos paralelo PanFS, utilizado no cluster de alto desempenho Netuno, instalado na UFRJ. São apresentados os princípios gerais da arquitetura de sistemas de arquivos baseados em objetos, em particular a do PanFS, a configuração do Netuno, e os resultados de testes que medem a eficiência do sistema de arquivos através de taxas de leitura e escrita.

1. Introdução

Na definição e configuração de um *cluster* há uma série de opções arquiteturais disponíveis, que devem ser cuidadosamente definidas para que esse *cluster* possa oferecer a eficiência e o desempenho esperados. Por exemplo, deve haver uma escolha harmoniosa de processadores, configuração dos nós, redes de interconexão e de E/S, conjunto de *softwares* oferecidos e ferramentas de administração para que esses objetivos sejam atingidos. [1]

Frequentemente a definição do sistema de armazenamento fica em segundo plano, tendo em vista que o perfil das aplicações de computação de alto desempenho é, em geral, de uso intensivo de processador. Há uma percepção equivocada de que a demanda dos nós computacionais sobre o sistema de arquivos não seja significativa.

No entanto, ao considerar-se um *cluster* como um todo, e não cada nó em separado, uma quantidade expressiva de dados pode ser produzida e consumida em um curto espaço de tempo. Segundo Ken Batcher, "um supercomputador é um dispositivo para transformar problemas *cpu-bound* em problemas *I/O-bound*".[5] O acesso a um sistema de arquivos remoto e compartilhado pode ainda causar atrasos ou problemas de sincronismo, que podem impactar de modo negativo o desempenho das aplicações.

O objetivo desse artigo é mostrar a configuração e a avaliação de desempenho do sistema de arquivos PanFS, utilizado no *cluster* Netuno.[1] Para isso, apresentamos na Seção 2 uma breve introdução aos sistemas de arquivos orientados a objetos, e na seção seguinte detalhes do PanFS. A Seção 4 mostra um resumo da configuração global do Netuno. As ferramentas utilizadas para medir as taxas de transferência de arquivos são descritas na Seção 5, e os resultados da avaliação de desempenho na Seção 6. Finalmente, as conclusões podem ser encontradas na seção seguinte.

2. Sistemas de Arquivos Orientados a Objetos

Para que os diversos nós de um *cluster* possam acessar os mesmos dados, é necessário um sistema de arquivos que apresente escalabilidade suficiente para atender requisições de um elevado número de clientes, simultaneamente e com atraso mínimo. Além disso, é necessário também um controle rígido sobre os metadados de forma a garantir o sincronismo e a coerência.

Escolher o sistema de arquivos adequado às necessidades de E/S previstas para as aplicações a serem executadas no Netuno foi uma das grandes dificuldades enfrentadas durante sua especificação.[1]

Os sistemas em que um servidor de arquivos é acessado pela rede usando o padrão NFS (*Network File System*) foram bastante utilizados por *clusters* por algum tempo. Todavia, tais opções não apresentam a escalabilidade desejada, quer seja em capacidade ou em desempenho. [2]

Nesse modelo, o de um sistema distribuído, todas as requisições de E/S passam por um único servidor, que fica responsável por alocar *inodes*, gerenciar *locks,* localizar blocos e ainda encaminhar os dados para os dispositivos de armazenamento. Cria-se assim um gargalo no acesso, que se torna mais grave à medida que um número maior de nós tentam acessá-lo.

Para que o sistema de arquivos possa alcançar a escalabilidade necessária, é preciso eliminar esse gargalo. Isso significa que os acessos devem poder ocorrer paralelamente aos servidores de dados sem a interferência de servidores centrais [3].

Recentemente, uma abordagem diferente levou ao desenvolvimento de uma nova geração de sistemas de

arquivos, paralelos e orientados a objetos. O princípio desses sistemas é que os canais de dados e metadados passem a ser separados, e os clientes possam transmitir seus dados diretamente para os dispositivos de armazenamento, sem a intervenção dos servidores de metadados. [2]

Nos sistemas orientados a objetos, o cliente ainda acessa inicialmente o gerenciador de metadados, entretanto, ao invés de mediar todo o processo, este apenas gera um mapa com a localização dos dados e entrega-o de volta ao cliente, permitindo que o mesmo troque informações diretamente com os dispositivos.

Entre as principais soluções de sistemas paralelos orientadas a objetos utilizadas correntemente estão o Lustre, fornecido pela Sun Microsystems, o PVFS, de código aberto, e o PanFS, da Panasas, Inc. As três opções foram consideradas para uso no Netuno, sendo que a decisão foi pelo PanFS, que pareceu ser o mais adequado à demanda prevista. [1]

Cabe ressaltar que está em desenvolvimento o padrão pNFS, que adiciona capacidade de paralelismo ao NFSv4.[10] A Panasas vem tentando manter o PanFS o mais próximo possível desse padrão [6], o que pode ser um indicativo de compatibilidade com futuros sistemas.

3. PanFS

O PanFS é um sistema de arquivos baseado em objetos cuja proposta é garantir altos níveis de desempenho e escalabilidade. Para isso, se utiliza de um *hardware* dedicado, vendido em conjunto com o sistema. [3]

A unidade básica é uma gaveta de altura 4U, para montagem em gabinete. Essa gaveta acomoda até 11 *blades*, sendo que um deles é obrigatório e tem o papel de um servidor de metadados, que nesse sistema é chamado de *Director Blade* (cartão diretor). Os outros são *Storage Blades* (cartões de armazenamento), onde os dados ficam efetivamente armazenados. [6]

Cada cartão de armazenamento é um sistema completo, com placa mãe, processador, memória, duas interfaces Gigabit Ethernet e dois discos de 500GB cada, totalizando 1TB por cartão, ou 10TB por gaveta. Sua memória é usada também como cache para os dados trocados.

Para serem armazenados, os arquivos são fracionados em blocos de 64KB. Os atributos do arquivo, um identificador e outros metadados são adicionados para formar um objeto [2]. Os objetos são então escritos em cartões independentes, usando RAID-5 para arquivos de tamanho superior a 64KB, ou RAID-1 para arquivos com apenas um objeto, ou seja, arquivos de até 64KB. Ao fazer o RAID com base nos arquivos, e não nos blocos físicos do disco, o PanFS pode atingir um alto grau de paralelismo, melhorando seu desempenho.

O protocolo de acesso a dados, chamado *DirectFlow,* implementa o modelo proposto para os sistemas baseados em objetos - toda vez que um nó do *cluster* faz uma requisição de E/S, o cartão diretor envia a ele um mapa com a localização dos objetos nos cartões de armazenamento, e depois deixa que o nó se comunique diretamente com eles. No caso de uma escrita, o cartão diretor é responsável por determinar a localização dos objetos nos cartões de armazenamento, tomando como base regras de *striping*, RAID, e balanceamento de carga.

Além do modo DirectFlow, se necessário, o cartão diretor também pode agir como um servidor NFS ou CIFS comum, intermediando o acesso aos cartões de armazenamento como simples dispositivos. [6]

4. Configuração do Netuno

O *cluster* Netuno é composto de 256 nós computacionais, cada um contando com dois processadores Intel E-5430 de 64 bits, memória cache de 12 MB e quatro núcleos por processador. [1] Isso significa que são 2048 núcleos, potencialmente clientes do sistema de arquivos, além dos 4 nós de acesso, de configuração semelhante aos nós computacionais, através dos quais os usuários têm acesso a seus diretórios, montados no PanFS.

Duas redes conectam os nós - uma rede Infiniband 20 Gbps, não-bloqueante, *full-duplex*, se destina à comunicação entre processos, enquanto o acesso ao sistema de arquivos é realizado por uma rede *ethernet*, cuja interface em cada nó é compartilhada pelos oito núcleos que compõem os dois processadores.

A organização dessa rede ocorre da seguinte forma: cada 32 nós se interligam a um *switch* local, com 48 portas Gigabit Ethernet, e cada *switch* se conecta através de um par de fibras óticas de 10 Gbps até um "*core switch*" Cisco 6509. Este está ligado ao sistema de armazenamento Panasas através de 12 canais *ethernet* de 1Gbps cada. Essa escolha não foi a desejada, contudo no momento da especificação do Netuno ainda não estavam disponíveis opções de melhor desempenho, como a 10GE [1].

O armazenamento em si é composto de 3 prateleiras, ou seja, 30 cartões de armazenamento, totalizando 30TB, sendo 27 TB úteis, devido ao espaço perdido na configuração do RAID, e mais 3 cartões de diretório. Além disso, cada nó computacional conta com um disco local SATA de 160GB, usado principalmente para efetuar operações de *swap* e armazenar uma cópia local do sistema operacional.

5. Método Experimental

Avaliar o desempenho do sistema de arquivos não é uma tarefa trivial. Ao medir-se o desempenho, certos cuidados devem ser tomados. Por exemplo, o uso de *buffers* e da memória *cache,* ou a busca antecipada de dados, podem levar a resultados que aparentam ter vazão maior do que a máxima esperada. Por outro lado, se um dos canais estiver congestionado, possivelmente um *switch*, os resultados podem aparentar um desempenho ruim que não necessariamente deve ser atribuído ao sistema de arquivos.

Com a configuração do Netuno, para transitar entre um nó e o disco, os dados passam por diversos pontos intermediários, influenciando o desempenho final - o sistema operacional do nó, o *switch* do gabinete, o *switch* central, o cartão de diretório, e, por fim, a memória do cartão de armazenamento.

As ferramentas usadas para *benchmarking* foram o Iozone [8] e o MPI-IO Test [7]. A primeira é uma ferramenta de avaliação de sistemas de arquivos comuns, adaptada para ser executada em *clusters* e gerar um *throughput* agregado via rsh ou ssh. Já a segunda utiliza a biblioteca MPI-I/O, que permite a vários processos, executando em nós diferentes, abrir e compartilhar arquivos remotos de maneira consistente. Nesse aspecto, uma operação pode ser N:N, quando os processos escrevem em arquivos independentes, ou N:1, quando todos os processos escrevem em um mesmo arquivo compartilhado [4]. Além disso, o teste permite comparar as diversas formas de acesso aos arquivos oferecidas pela biblioteca MPI I/O.

O fator mais importante a se considerar é a influência que o sistema de arquivos terá sobre o desempenho de aplicações reais executadas no *cluster*. Nesse caso, deve-se levar em consideração o padrão de acesso das mesmas. O uso desses *benchmarks* serve apenas como uma estimativa, já que cada aplicação tem seu próprio perfil, que raramente será tão padronizado quanto o das ferramentas de *benchmark.*

O primeiro teste realizado é o de throughput agregado, na ferramenta Iozone. Obtivemos uma taxa de **422MB/s** para a escrita seqüencial e **1.64GB/s** para a leitura. Esse é o desempenho máximo que pode ser esperado para uma dada aplicação.

6. Resultados

O primeiro teste realizado é o de throughput agregado, na ferramenta Iozone. Obtivemos uma taxa de **422MB/s** para a escrita seqüencial e **1.64GB/s** para a leitura. Esse é o desempenho máximo que deve ser esperado para uma dada aplicação que, como esse teste, não utilize o protocolo *DirectFlow*

No MPI-IO Test, comparamos a escrita com e sem o uso do *DirectFlow*, com um tamanho fixo de 32GB para o arquivo e 64MB para os blocos a serem escritos, grandes o suficiente para minimizar os efeitos dos *buffers*. Os resultados vistos na Figura 1 indicam uma convergência para uma taxa de escrita de **220 MB/s** sem o uso do protocolo e **700 MB/s** com o mesmo.

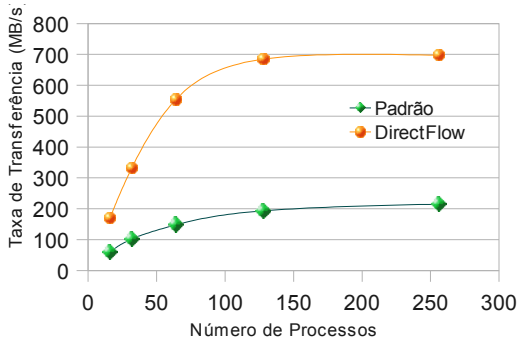

Figura 1: Taxa de Escrita do PanFS

A seguir, avaliamos a diferença da escrita N-N para a escrita N-1, considerando a diferença de desempenho para diferentes tamanhos de arquivo e de bloco escrito.

Usamos dois casos de teste: no primeiro caso mantivemos os tamanhos de 32GB e 64MB e no segundo caso usamos os tamanhos de 4MB para o arquivo e 16KB para o bloco, para verificar os efeitos da escrita de blocos pequenos sobre o RAID do PanFS. Os resultados desses testes são sumarizados nas Figuras 2 e 3, respectivamente.

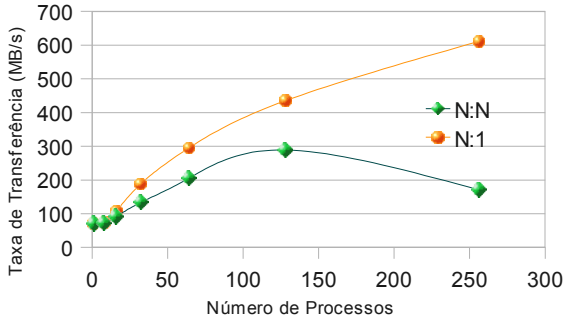

Figura 2: Taxa de escrita para arquivos de 32GB

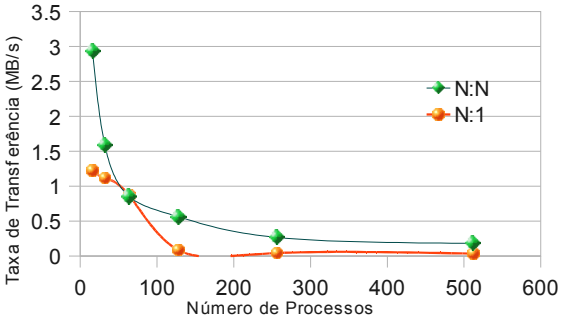

Figura 3: Taxa de escrita para arquivos de 4MB

Os mesmos testes foram aplicados para avaliar a leitura, um para arquivos de 32G (blocos de 64MB) e outro para arquivos com tamanho muito menor, de apenas 64MB (blocos de 128KB), cujos resultados podem ser vistos na Figura 4.

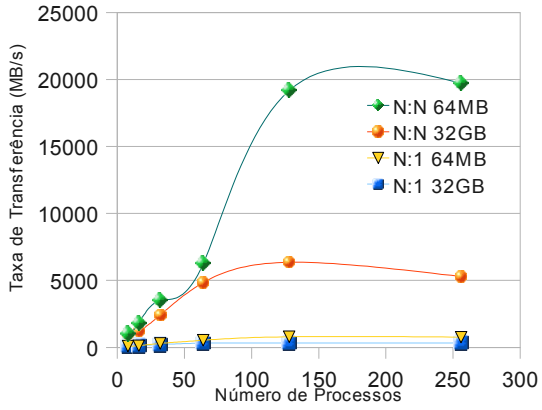

Figura 4: Taxas de Leitura para 64MB e 32GB

Comparamos ainda o desempenho da escrita para o caso em que só um processo está ativo por nó com o caso em que 8 processos estão ativos por nó, um em cada núcleo, de forma a avaliar o efeito do compartilhamento da interface de rede. Foram usados 64 processos nesse teste, e arquivos de 32GB. Os resultados são mostrados na Figura 5.

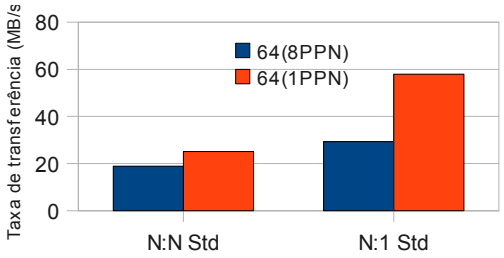

Figura 5: Taxa de escrita para 64 processos

Avaliamos, por fim, a comparação de chamadas coletivas e independentes, com e sem padrão de *striding*, observando pouca variação dos resultados. Possivelmente, a influência desses fatores seria mais facilmente observada em uma aplicação, onde os padrões de escrita são menos uniformes

7. Conclusões e Trabalhos Futuros

Os resultados dos testes mostrados na Figura 1 comprovam que, de fato, o protocolo DirectFlow proporciona um ganho sensível no desempenho da escrita concorrente.

O teste de escrita em arquivos de 4MB, exibido na Figura 3, mostra que o sistema não lida bem com a escrita em blocos muito pequenos, possivelmente por causa de suas regras de RAID específicas, e da maior frequência de colisões causadas pelas requisições chegando simultaneamente aos *switches*.

Alguns testes de leitura da Figura 4, assim como o teste de *throughput* do Iozone, excedem aparentemente a taxa máxima de transferência da rede (12 x 1Gbps = 1.5GBps), o que evidencia o efeito dos *buffers* observado na Seção 5.

É possível também notar que a taxa de leitura tem valores significativamente maiores quando os processos leem arquivos independentes do que no caso da leitura em um arquivo compartilhado, visto que o segundo caso depende não somente da taxa de transferência da rede e do sistema de arquivos como também da concorrência no acesso ao arquivo lido.

No entanto, no teste de escrita na Figura 2, quando o número de processos aumenta, a escrita em arquivos independentes tem desempenho inferior à escrita em um arquivo compartilhado, contrariando a expectativa observada em outros sistemas [9]. Isso evidencia um possível *overhead,* ressaltado no teste seguinte, que mostra que, quando os processos são alocados a nós diferentes, a escrita N:1 melhora, mas não a N:N, indicando a possibilidade de que a limitação esteja na rede, e não no sistema de arquivos.

Pretendemos realizar ainda uma análise detalhada para identificar a causa dessa limitação. Conforme mencionado, tentaremos também avaliar o desempenho usando não somente os padrões fixos das ferramentas de *benchmark*, mas os fluxos de entrada e saída de aplicações reais.

Além disso, iremos implementar outras opções de sistemas de arquivos orientados a objeto e comparar o desempenho com o do PanFS.

8. Referências

[1] Vinícius Silva, Cristiana Bentes, Sérgio Guedes, Gabriel P. Silva, "Arquitetura e Avaliação do Cluster de Alto Desempenho Netuno". Submetido ao WSCAD-SSC 2009.

[2] Jeffrey B.Layton, "Parallel Platters: File Systems for HPC Clusters". Linux Magazine, Novembro 2007.

[3] Brent Welch, Marc Unangst, "Clustered and Parallel Storage System Technologies". SC07 / Fast09.

[4] James Nunez, John Bent, "MPI-IO Test User Guide".

[5] Ken Batcher, "Design of a Massively Parallel Processor". IEEE Transactions on Computers, Vol. C29, September 1980, 836-840.

[6] Panasas. http://www.panasas.com

[7] MPI-I/O Test. http://public.lanl.gov/jnunez/benchmarks/mpiiotest.htm

[8] Iozone. http://www.iozone.org/

[9] Gary Grider, Hsing-bung Chen, James Nunez et. al., "PaScal – A New Parallel and Scalable Server IO Networking Infrastructure for Supporting Global Storage/File Systems in Large-size Linux Clusters".

[10] pNFS. http://www.pnfs.com/