1
Introdução

A computação em grade se caracteriza pelo uso de recursos computacionais distribuídos em várias redes. Os diversos nós contribuem com capacidade de processamento, armazenamento de dados ou quaisquer outros recursos disponíveis. Nesse cenário, diversos problemas precisam ser tratados, tais como a heterogeneidade de plataformas, a forma de compartilhamento dos dados, os mecanismos de agendamento de processos e a segurança no acesso aos recursos.

A execução de aplicações na grade requer uma ampla disponibilidade de dados. Considerando uma abordagem orientada a tarefas, usada em diversas aplicações científicas, um programa precisa processar uma entrada gerando uma saída que será posteriormente analisada por um usuário, ou fornecida como entrada para outra tarefa. Nos casos em que a aplicação é composta por uma série de programas encadeados, ao tentar executar esse fluxo de programas em vários computadores, um mecanismo de compartilhamento de dados é essencial para permitir que todos eles tenham acesso aos dados.

Diversas iniciativas para o gerenciamento e transferência de arquivos foram apresentadas pela comunidade [1, 2, 3, 4, 5]. Cada um desses trabalhos foi desenvolvido de acordo com as características dos sistemas e da rede onde eles seriam utilizados. A existência desse esforço, nos consagrados sistemas como o Globus [6], Condor [7] e Google[1], sugere a necessidade de um sistema de gerenciamento específico para o cenário de computação em grade e ambientes distribuídos.

Uma abordagem comum consiste em permitir que os arquivos sejam transferidos entre as diversas máquinas usando o FTP como protocolo de transferência [1, 3]. Dessa forma, os arquivos são replicados e podem ser utilizados em diversos pontos da grade. Uma infra-estrutura de servidores FTP deve ser montada de forma a permitir a cópia dos arquivos entre as diversas

[1] http://www.google.com

máquinas e um mecanismo de endereçamento é necessário para viabilizar a localização e controle dos arquivos.

Uma solução mais flexível, que permite o acesso remoto aos arquivos sem a necessidade de cópias entre as diversas máquinas, pode ser obtida através do uso de sistemas de arquivos distribuídos tradicionais, como o NFS [8] e o AFS [9]. Nesses sistemas, diversas máquinas podem disponibilizar o serviço e uma federação deste pode ser construída com a utilização de redirecionamentos entre eles. A localização do arquivo é dada pelo seu caminho, com base na raiz do sistema.

De acordo com a aplicação a ser executada, pode ser preferível o acesso remoto via NFS, como no caso em que apenas uma parte do arquivo é acessada, ou uma transferência completa pode ser mais interessante, para os casos onde o arquivo é utilizado múltiplas vezes e um eventual mecanismo de *cache* do sistema de arquivos não é capaz de armazenar o arquivo completamente no cliente. Como veremos mais adiante, também existem situações onde a disponibilização dos servidores NFS gera esforços administrativos relativamente altos, fazendo com que a cópia dos arquivos seja tomada como uma alternativa mais simples.

Este trabalho propõe o GridFS, um sistema que visa combinar alguns benefícios das abordagens baseadas em servidores FTP com algumas características dos sistemas de arquivos distribuídos tradicionais, bem como oferecer funcionalidades especiais para computação em grade. Em uma única infra-estrutura, e inspirados nas características dos sistemas e ambientes descritos anteriormente, oferecemos três conjuntos principais de funcionalidades: 1) baseado no uso de servidores FTP como mecanismo de cópia de arquivos, disponibilizamos funções que permitem a cópia eficiente de arquivos entre as diversas máquinas que possuem o GridFS; 2) considerando os sistemas de arquivos distribuídos, oferecemos uma *API* que permite a navegação na árvore de arquivos, assim como o acesso remoto aos mesmos, o que viabiliza o uso dos arquivos pelas aplicações sem a necessidade de cópias para uma área local; e 3) levando em consideração os ambientes de computação em grade, fornecemos um suporte a metadados e definimos funcionalidades que provêem estimativas do tempo de transferência de arquivos entre as máquinas. Ou seja, em um único sistema para o gerenciamento e compartilhamento de arquivos distribuídos, combinamos os principais benefícios das abordagens descritas anteriormente e disponibilizamos um suporte mais específico para aplicações de computação em grade.

Apesar do controle de acesso ser um item de grande importância para um servidor de arquivos, a versão atual do sistema não incorpora o conceito de usuários e permissões. Todos os arquivos são armazenados no sistema de arquivos local dos servidores e qualquer processo que tenha acesso ao GridFS é capaz de realizar as operações de cópia, criação ou remoção de arquivos. O acesso aos servidores deve ser controlado através de mecanismos externos ou usando mecanismos de controle de acesso definidos em CORBA [10, 11].

Na seção 1.1, apresentamos as características do sistema e reiteramos as contribuições desse trabalho na seção 1.2. Em seguida, apresentamos a estrutura restante deste documento na seção 1.3.

1.1 Características

O desenvolvimento do GridFS teve como motivação inicial atender as necessidades de algumas aplicações desenvolvidas no nosso grupo de pesquisa. Mais especificamente, queríamos utilizá-lo como mecanismo de gerenciamento de arquivos nos projetos CSBase [12] e ActivePresentation [13, 14]. O CSBase é uma infra-estrutura para *clusters* e grades computacionais, e o ActivePresentation é uma infra-estrutura para execução de apresentações multimídia distribuídas.

De acordo com as necessidades dessas aplicações, e após a realização de um estudo sobre os principais benefícios das soluções disponíveis para gerenciamento de arquivos distribuídos, definimos uma série de características que o GridFS deveria possuir, visando facilitar o compartilhamento de arquivos em grades e, de forma mais genérica, em ambientes distribuídos heterogêneos. As seguintes subseções apresentam cada uma dessas características.

Escalabilidade, Interoperabilidade e Desempenho

Considerando a capacidade de armazenamento de uma grade computacional, da ordem de terabytes, a criação de uma federação de servidores deve ser possível, de forma que o número de arquivos gerenciados pelo sistema seja dado pelo total de arquivos armazenados em cada um dos servidores. Para viabilizar o gerenciamento lógico dos dados, a criação da federação pode ser

realizada com a definição de *mount points*, permitindo a um determinado local em um servidor referenciar um outro local na federação.

Devido à heterogeneidade das máquinas e sistemas operacionais, o sistema deve trabalhar sobre diversos formatos e particionamentos de discos, bem como ter a capacidade de ser executado em diversas plataformas. Aliando-se essa característica à possibilidade de criação de uma federação de servidores, o sistema deve ser capaz de integrar diversos sistemas de arquivos em uma única árvore distribuída.

Em aplicações de alto desempenho, o fator limitante da computação pode estar relacionado a diversas características do ambiente onde a aplicação está em execução. Tipicamente, as aplicações são limitadas pela *CPU*, disponibilidade de memória *RAM*, velocidade de acesso à memória secundária ou velocidade da troca de mensagens pela rede. Considerando as operações de entrada e saída de dados do GridFS, o fator limitante do seu desempenho deve estar relacionado com a capacidade de entrada e saída do servidor onde ele está em execução, ou seja, com a velocidade de acesso ao disco e rede.

Interfaces

A interface programática do sistema deve ser simples e possuir métodos agrupados segundo as funcionalidades oferecidas. Por exemplo, o acesso a um diretório remoto deve permitir a navegação pelos seus filhos ou a criação de novos elementos. A obtenção de um canal de acesso a um arquivo deve permitir a leitura ou escrita de dados remotamente, sem a necessidade de cópia. O usuário também deve ser capaz de comandar as cópias de arquivo diretamente entre dois servidores remotos, sem que ele precise servir de ponte para a transferência dos dados. Além da *API* para acesso aos arquivos, o sistema deve oferecer mecanismos que viabilizem o acesso aos dados por parte de aplicações legadas.

Informações Auxiliares

Um mecanismo de suporte a metadados deve estar presente, permitindo o armazenamento de informações sobre os arquivos. Os metadados facilitam a utilização de um sistema auxiliar para a consulta por arquivos, permitindo que sistemas de indexação e busca sejam mais facilmente implantados. Além disso,

estimativas sobre o tempo de transferência de arquivos entre os servidores devem estar presentes. Algumas aplicações podem requerer um processamento intenso sobre um pequeno conjunto de dados, enquanto outras aplicações podem utilizar algoritmos que tratam uma grande quantidade de dados mas não requerem muito processamento. Dessa forma, as estimativas podem servir como fonte de informação para que escalonadores de processos julguem a conveniência da cópia de um arquivo, levando em consideração o custo de transferência , as características das aplicações e a disponibilidade dos recursos.

Independência, Coesão e Simplicidade

Para que o sistema seja facilmente implantado e utilizado por um grande conjunto de pessoas e ambientes, ele não deve depender da existência de uma infra-estrutura complexa ou requerer um conhecimento altamente específico para sua instalação e manutenção. Ele deve ser autocontido, de forma a tratar o problema de gerenciamento de arquivos com poucas dependências de componentes externos. Além disso, ele deve ser simples e ter o seu desenvolvimento guiado por um conjunto de funcionalidades básicas amplamente sedimentadas e com semânticas bem definidas.

Tolerância a Falhas e Segurança

O sistema deve ser capaz de suportar falhas de comunicação, *hardware*, ou sobrecargas nos servidores. Em situações extremas, os clientes devem receber mensagens de erro elucidativas e, no momento do reestabelecimento do *link* ou servidor, o cliente deve voltar a operar normalmente.

O acesso aos arquivos deve ser limitado às aplicações que possuem autorização para a manipulação da árvore distribuída. Essa característica pode ser atingida utilizando um mecanismo de autenticação do usuário, ou restringindo o acesso remoto ao servidor para algumas máquinas específicas, através do uso de mecanismos externos ao GridFS.

1.2 Contribuições

O objetivo dessa dissertação é definir e implementar um servidor de arquivos, o GridFS, capaz de atender as necessidades de compartilhamento de arquivos em grades e ambientes distribuídos heterogêneos, levando em consideração os diversos aspectos apresentados na seção 1.1. O sistema inova por agregar todas as características em uma única infra-estrutura portável, interoperável, escalável, independente, e de fácil instalação e manutenção, provendo mecanismos de transferência de dados com desempenho equivalente ao FTP, mecanismos de acesso a dados remotos semelhante aos sistemas de arquivos distribuídos, e funções especializadas para ambientes de computação em grade. Além da definição e implementação do GridFS, esta dissertação também apresenta um estudo experimental sobre a escalabilidade de alguns mecanismos de gerenciamento de objetos remotos e sobre o desempenho de diversos métodos de cópia de arquivos.

O *framework* CSBase [15], um dos projetos que motivou este trabalho, oferece uma infra-estrutura para *clusters* e grades computacionais e está sendo utilizado em vários projetos, junto a parceiros industriais, contando com a contribuição de centenas de usuários e dezenas de desenvolvedores. Como forma de avaliação e aplicação do GridFS, o sistema foi integrado ao CSBase e permite a disponibilização e cópia de arquivos nas diversas máquinas gerenciadas pelas instâncias do *framework*.

Um dos objetivos da integração é reduzir a dependência do CSBase em relação ao sistema de arquivos distribuído atualmente em uso, e assim facilitar sua implantação em ambientes heterogêneos compostos por centenas de máquinas de diferentes plataformas. Incorporando o GridFS no CSBase, evitaremos a necessidade da configuração específica dos sistemas de arquivos nas diversas máquinas, fazendo com que a instalação do CSBase leve consigo uma solução própria, pré-configurada, para o compartilhamento de arquivos. O uso do GridFS também facilitará que o *framework* seja empregado além dos domínios administrativos de uma mesma instituição.

No CSBase, para que um programa seja disparado em um nó de execução, a máquina deve possuir acesso ao binário e aos dados a serem processados. A integração do GridFS consiste na utilização da técnica de *staging*, onde os binários e os arquivos de entrada são transferidos para a máquina de execução e, após o término da execução, os arquivos contendo o resultado são transferidos

para a área de armazenamento. A utilização das demais funcionalidades do GridFS no CSBase, tais como a *API* para acesso remoto e as estimativas de desempenho da cópia, foram projetadas para atender novas demandas das aplicações.

As necessidades do CSBase indicaram a importância do GridFS e serviram como guia para o nosso desenvolvimento. Contudo, apesar de ter o CSBase como usuário direto, o sistema foi desenvolvido de maneira independente e pode ser utilizado nas mais diversas aplicações que necessitem da manipulação e gerenciamento de arquivos remotos, especialmente em ambientes heterogêneos.

1.3 Estrutura da Dissertação

O restante desse documento está organizado da seguinte forma: o Capítulo 2 apresenta alguns trabalhos relacionados e realiza algumas considerações sobre os sistemas apresentados. O Capítulo 3 apresenta o GridFS e discute diversos aspectos que foram avaliados durante a sua modelagem e implementação, levando em consideração as características anteriormente especificadas. O Capítulo 4 apresenta diversos resultados obtidos nos testes de desempenho e escalabilidade do serviço. O Capítulo 5 mostra a arquitetura do CSBase, apresentando os novos componentes criados com base no GridFS e discute a integração dos dois sistemas. Finalmente, o Capítulo 6 apresenta as conclusões e enumera alguns trabalhos futuros que podem ser realizados para ampliar as funcionalidades do GridFS ou atingir outros cenários de uso.