Obtenção de Tolerância a Falhas na ferramenta de computação MyGrid *

Hélio Antônio Miranda da Silva†, Carolina Ming Chiao‡

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul
Caixa Postal 15064 – 90501-970 Porto Alegre, RS

{hamsilva,cchiao}@inf.ufrgs.br

Resumo. *O avanço tecnológico permitiu o surgimento de um novo conceito que é o da computação em Grid e de ferramentas como o MyGrid que dão o suporte necessário para a execução de aplicações nestes ambientes. Este trabalho visa apresentar a utilização da recuperação por retorno como mecanismo para aumentar a disponibilidade e a confiabilidade da execução de tarefas bag of tasks em ambientes de computação em Grid utilizando a ferramenta MyGrid.*

1. Introdução

Nos últimos tempos, os avanços tecnológicos consideráveis que ocorreram na computação, como o aumento na velocidade de processamento, o aumento na capacidade de armazenamento e alto desempenho das redes de comunicação, permitiram o surgimento de um novo conceito que é o da computação em Grid. A computação em Grid [Foster et al., 2001] tem emergido como uma área de pesquisa importante por permitir o compartilhamento de recursos computacionais geograficamente distribuídos entre vários usuários. Assim, a computação em Grid está relacionada ao compartilhamento coordenado de recursos e a solução de problemas em uma organização virtual dinâmica, multi-institucional [Foster, 2002].

A computação em Grid permite a criação de organizações virtuais com o compartilhamento de recursos computacionais e de dados, garantindo um acesso controlado e seguro a recursos como capacidade de processamento paralelo e de armazenamento. Todo o acesso de usuários a esses recursos, que estão geograficamente espalhados, é realizado através de uma visão unificada do sistema, isto é, os recursos compartilhados passam e ser vistos como um grande computador virtual. Também podem executar aplicações que demandam grande poder de processamento, compartilhar informações e permitir colaboração entre diferentes instituições. A computação em Grid pode ser utilizada para vários objetivos como, por exemplo, na obtenção de alto desempenho em aplicações que exigem uma grande demanda computacional como simulações climáticas, simulações de fenômenos astrofísicos ou em computações científicas em geral. Outros exemplos de áreas onde as tecnologias Grid podem ser aplicadas são a computação *peer-to-peer* (P2P) e a criação de ambientes virtuais colaborativos [Brunett et al., 1998].

O desenvolvimento de aplicações e ferramentas em ambientes de computação em Grid tem sua complexidade aumentada pela heterogeneidade e pelas freqüentes mudanças que ocorrem nestes ambientes em função do comportamento dinâmico dos seus recursos. Visando amenizar este problema, existem ferramentas como o *Globus Toolkit*

*Parcialmente financiado pelo projeto DepGriFE/HP Brasil P&D
†Bolsista financiado pela CAPES.
‡Bolsista financiada pelo projeto DepGriFE/HP Brasil P&D

[Foster and Kesselman, 1998] que é um conjunto de componentes de *software* projetados para dar suporte ao desenvolvimento de aplicações que serão executadas em ambientes de computação em Grid. Outro exemplo é a ferramenta *MyGrid* [Cirne et al., 2003b] onde as tarefas são lançadas e coordenadas através de um nó chamado *Home*, seguindo o modelo de computação *bag of tasks* [Cirne et al., 2003a], em que aplicações paralelas são compostas por tarefas independentes umas das outras.

Estes ambientes tratam de uma imensa variedade de recursos espalhados entre vários domínios, tornando o sistema extremamente heterogêneo. A execução de aplicações em ambientes Grid pode envolver centenas e até milhares de nós. Assim, a probabilidade de algum recurso falhar torna-se alta, já que falhas em nós e desconexões tornam-se muito freqüentes [Medeiros et al., 2003]. Por isso, a tolerância a falhas torna-se uma questão chave. Então, para que se tenha um maior rendimento, é necessário que estas plataformas apresentem uma confiabilidade e uma disponibilidade adequadas. Assim, torna-se necessário fazer com que o ambiente se recupere destas falhas e consiga completar sua execução ao mesmo na presença de falhas através de mecanismos que não prejudiquem significativamente o desempenho do sistema[Bosilca et al., 2002].

Este trabalho tem por objetivo definir o funcionamento de um mecanismo de recuperação através de *checkpoints*, que será utilizado no nó central da ferramenta *MyGrid*. Este nó central é responsável pelo escalonamento de tarefas e pelo gerenciameno do sistema. A ferramenta *MyGrid* é uma proposta de plataforma Grid desenvolvida na Universidade Federal de Campina Grande (UFCG). Na atual implementação do *software MyGrid*, existe uma máquina que centraliza o gerenciamento e o escalonamento chamada máquina *Home*. A ocorrência de falhas na máquina *Home* acarreta na perda das aplicações que estavam sendo executadas no Grid, podendo resultar em horas ou até mesmo dias de processamento perdido. Este trabalho, portanto, apresenta a implementação de uma solução para este problema, em que após a ocorrência de uma falha na máquina *Home*, uma nova máquina *Home* será lançada com o último estado consistente do escalonador.

Este trabalho mostra como está sendo realizada a implementação do mecanismo de recuperação no *MyGrid*. Na seção 2 são apresentados o modelo de sistema e o modelo de falhas adotado. Na seção 3 é definido, em linhas gerais, o funcionamento do *MyGrid*. Na seção 4 é mostrado o funcionamento do mecanismo de recuperação. E, por último, na seção 5 são apresentadas as conclusões referentes ao trabalho.

2. Modelo do Sistema

Um Grid pode ser composto por um conjunto de máquinas que estão espalhadas através de muitos domínios. O modelo de computação que irá ser utilizado neste trabalho é o modelo de computação *bag of tasks*. Nele existe um conjunto de tarefas no qual a execução de cada uma destas tarefas é completamente independente das demais, o que implica na inexistência de comunicação entre estas tarefas durante as suas execuções. Também é importante salientar que neste modelo de computação, as tarefas podem ser realizadas em qualquer ordem [Cirne et al., 2003a]. A ferramenta *MyGrid*, que servirá como base de estudos neste trabalho, utiliza este modelo de computação *bag of tasks*.

Quanto ao modelo de falhas, é adotado um modelo de *colapso*, onde, se uma tarefa que está sendo executada em uma máquina for considerada como em falha, toda a sua execução é perdida. A máquina em que esta tarefa estava sendo executada, a princípio não é excluída do Grid, pois a falha gerada durante a execução pode ter sido gerada por problemas no ambiente de execução ou por problemas de configuração da tarefa. Assim, outras tarefas poderão ser lançadas neste nó e ter a sua execução concluída com sucesso, já que existe a probabilidade de que este problema não volte a se repetir. Caso o nó torne-

se inatingível, devido a um *crash* na máquina ou devido a falhas na rede de comunicação, este nó não terá mais tarefas alocadas a ele e passará a ser considerado fora do sistema Grid.

3. Estrutura da ferramenta MyGrid

O *MyGrid* é uma ferramenta que tem por objetivo dar suporte à utilização de recursos em um Grid computacional. Ela visa contornar as dificuldades de gerenciamento e de escalonamento de tarefas em ambientes de computação em Grid. Esta ferramenta permite a execução de aplicações que seguem o modelo de computação *bag of tasks* nestes ambientes. Neste modelo, as tarefas são independentes umas das outras; além disso, a ordem em que são executadas não altera o resultado da computação. A total independência entre as tarefas faz com que não exista comunicação entre elas durante a computação [Cirne et al., 2003a]. A ferramenta *MyGrid* é implementada na linguagem de programação Java e em *scripts shell* e a sua implementação atual permite utilizá-la em várias versões do sistema operacional Linux.

3.1. Funcionamento do MyGrid

A estrutura do *MyGrid* está esquematizada na figura 1. A execução de uma aplicação paralela através do *MyGrid* é lançada através de um nó central do Grid que coordena o lançamento e o escalonamento das tarefas nas máquinas que realizam a computação. Este nó central é chamado de máquina *Home*. As outras máquinas que compõem a estrutura do Grid são chamadas máquinas Grid. Cada máquina *Grid* realizará uma, e somente uma, tarefa por vez. Toda comunicação existente dentro do sistema será iniciada pela máquina *Home*.

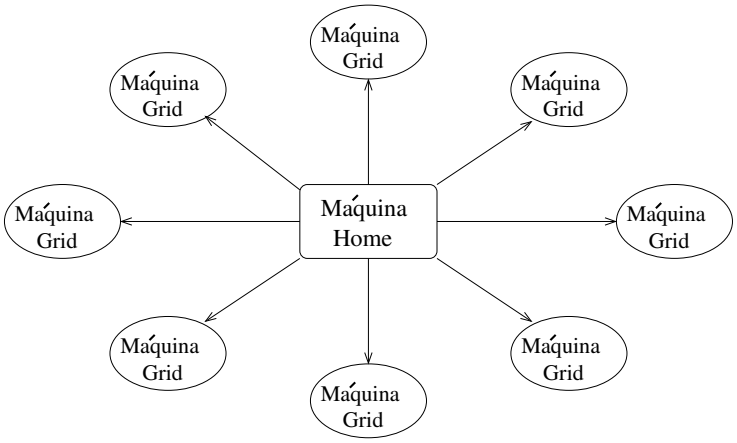

Figura 1: Estrutura do MyGrid

Existem três possibilidades de configuração para máquinas Grid que irão definir a estratégia de comunicação entre a máquina *Home* e as máquinas *Grid*. Na primeira delas, uma máquina Grid executa um processo chamado *UserAgent* que permite a comunicação do *MyGrid* com o nó. Esta é a estratégia de comunicação implementada pelo próprio *MyGrid*. Na segunda delas, as máquinas Grid possuem uma instalação do *Globus Toolkit*. Assim, o *MyGrid* realiza a interação com os serviços disponibilizados pelo *Globus*. E a última possibilidade de estratégia de comunicação é por tunelamento[1] utilizando o *software* SSH (*Secure Shell*).

[1]Palavra usada como tradução do termo *tunneling* do inglês.

3.2. Escalonamento de tarefas

O *MyGrid* possui, como suas principais entradas, um arquivo que define as máquinas que compõem o Grid e um outro arquivo que define todas as tarefas que devem ser executadas. Este conjunto de tarefas é chamado de *Job*, e cada uma das tarefas que compõem o *Job* é chamada de *Task*.

O escalonamento do *MyGrid* utiliza um algoritmo chamado *Workqueue with Replication* (WQR). Neste algoritmo, não é utilizado nenhum conhecimento a respeito do ambiente ou a respeito do tempo de execução das tarefas. No WQR, cada nova tarefa é escalonada para uma máquina *Grid* ociosa, enquanto existirem máquinas ociosas. A cada vez que uma máquina termina a execução ela volta a ser considerada ociosa e pronta para receber mais tarefas. Neste escalonamento, são criadas réplicas das tarefas com o objetivo de obtenção de desempenho [Cirne et al., 2003a]. Assim, para cada tarefa são lançados várias tarefas réplicas, e, após o retorno de qualquer uma das cópias da tarefa, todas as réplicas desta tarefa têm a sua execução interrompida. Mesmo considerando que o objetivo principal da replicação de processos é a obtenção de desempenho, este mecanismo também oferece tolerância a falhas, já que, mesmo que a execução de uma tarefa seja interrompida por intermédio de uma falha, a execução da aplicação paralela poderá continuar, pois possivelmente alguma das outra réplicas desta tarefa não será afetadas por esta falha e poderá continuar e concluir a execução desta tarefa em questão.

3.3. Máquina Home como um ponto único de falha

Na estrutura do *MyGrid*, a máquina *Home* apresenta-se como um ponto único de falha, pois caso uma falha ocorra neste nó a execução da aplicação paralela inevitavelmente será interrompida. Este trabalho propõe a realização de pontos de verificação (*checkpoints*) no estado da máquina *Home* como o intuito de sobrepor esta fragilidade do *software MyGrid*. Com isso, o estado da máquina *Home* poderá ser recuperado de forma consistente e a execução deste processo poderá ser continuada no mesmo nó ou em algum outro nó disponível no Grid.

4. Mecanismo de Recuperação

Com o objetivo de aumentar a disponibilidade da máquina *Home* do *MyGrid*, será utilizada a recuperação por retorno. A recuperação por retorno é uma técnica bem conhecida dentro da área tolerância a falhas, onde são realizados periódicos *checkpoints* de um processo. Estes *checkpoints* possuem as informações referentes ao estado deste dado processo. Com isso, a partir deste mecanismo, é possível retornar a execução de um processo a um estado anterior já computado toda vez que o sistema apresenta falha. Com isso, possíveis falhas no nó central do MyGrid poderão ser contornadas pelo mecanismo de recuperação, fazendo com que a disponibilidade de sistema seja aumentada, uma vez que este sistema permanecerá uma porção de tempo maior em funcionamento.

A implementação do mecanismo de recuperação será feita através de modificações e incrementações no código fonte do *MyGrid*. Isto é, serão inseridos os procedimentos necessários para que a própria máquina *Home* realize, em momentos necessários, o salvamento em arquivo das informações imprescindíveis para retomar a sua execução na ocorrência de falhas. Serão salvas somente as estruturas dados relevantes para a recuperação do contexto da execução da máquina *Home* permitindo, assim, a seu restabelecimento após qualquer falha.

Os *checkpoints* seriam salvos em um local considerado de armazenamento estável. Este local será em uma outra máquina do Grid ou em sistema de arquivos compartilhados. Assim, o estado do processo poderá ser recuperado em uma outra máquina.

4.1. Salvamento de estados da máquina Home

Conforme explicitado anteriormente, o salvamento de estado da máquina *Home* será realizado inteiramente em nível de aplicação através da modificações no código fonte do *MyGrid*. Nestas modificações são incluídos os procedimentos responsáveis pelo salvamento do estado do nó principal desta ferramenta. A princípio, as estruturas de dados que contêm as informações necessárias para definir o estado da máquina *Home* são as filas de tarefas a serem executadas, assim como os seus estados e as estruturas de dados referentes aos estados das máquinas que compõem o Grid computacional, isto é a composição do ambiente Grid. Portanto, com estas informações será possível recuperar o estado da máquina *Home* que é responsável pelo escalonamento e pelo lançamento das tarefas.

O processo de salvamento de estado será executado a cada mudança de estado do escalonador da máquina *Home*. Uma mudança de estado ocorre a cada nova tarefa lançada e também quando ocorre o retorno de alguma tarefa vindo de alguma réplica que estava sendo executada, seja este retorno realizado por intermédio de falhas ou por término da execução com sucesso. Isto é, cada vez que uma nova tarefa for lançada e a cada vez que ocorrer o retorno de uma execução, um novo salvamento de estado deverá ser realizado. Já que estes freqüentes salvamentos de estado poderiam tornar-se bastante onerosos para a execução do Grid, optou-se pela utilização de registros de *log* visando diminuir o impacto que os estes vários salvamentos de estado podem causar ao desempenho do sistema. Estes *logs* têm como objetivo armazenar mudanças ocorridas no processo da máquina *Home* entre dois salvamentos de estado. Com isso, os *checkpoints* serão realizados após realizados um determinado número de modificações no estado do processo *Home*, como, por exemplo, após a ocorrência de 10 modificações no estado da máquina *Home* será realizado um *checkpoint* e todos os *logs*, assim como, o antigo *checkpoint* serão excluídos. Assim, os registros de *logs* seriam utilizados para registrar os eventos que ocorreriam entre dois sucessivos *checkpoints*.

Um importante questão é quanto às conexões que estarão ativas quando o processo da máquina *Home* falhar. Para poder refazer estas conexões, é necessário que haja um controle de todas as conexões que estão abertas para que, em caso de falhas, estas conexões possam ser recriadas já sendo redirecionadas para a nova máquina *Home*, pois assim todos os nós que estão executando tarefas no Grid poderão retornar seus resultados para o novo processo que estará executando em uma nova máquina *Home*.

5. Conclusão

Dentro do conjunto de aplicações que podem se beneficiar da computação em Grid destacam-se as aplicações *bag of tasks*. O *software MyGrid* é uma plataforma que possui todo suporte necessário para a execução de aplicações deste tipo em ambientes de Grid. Contudo, a estrutura de funcionamento desta ferramenta apresenta a sua máquina *Home*, que centraliza o gerenciamento e o escalonamento de tarefas, como um ponto único de falha. Esta vulnerabilidade deste *software* merece atenção, pois falhas na máquina *Home* do *MyGrid* podem causar a perda de toda computação que estava sendo realizada no Grid. A recuperação por retorno é técnica bem conhecida dentro da área de tolerância a falhas que se propõe como uma boa solução para contornar esta fragilidade do ambiente de computação proposto pelo *MyGrid*. Assim, este trabalho expõe o projeto e a implementação de um mecanismo de recuperação por retorno neste nó central do *My-*

Grid, desta forma, aumentando a disponibilidade e a confiabilidade da utilização deste ambiente de computação.

Referências

Bosilca, G., Bouteiller, A., Cappello, F., Djilali, S., Fedak, G., Germain, C., Herault, T., Lemarinier, P., Lodygensky, O., Magniette, F., Neri, V., and Selikhov, A. (2002). MPICH-V: Toward a scalable fault tolerant MPI for volatile nodes. In *SC'2002 Conference CD*, pages 1–18, Baltimore, MD - USA. IEEE/ACM SIGARCH.

Brunett, S., Czajkowski, K., Fitzgerald, S., Foster, I., Johnson, A., Kesselman, C., Leigh, J., and Tuecke, S. (1998). Application experiences with the globus toolkit. In *Proceedings of 7th IEEE Symp. on High Performance Distributed Computing*, pages 81–89.

Cirne, W., Brasileiro, F., Sauvé, J., Andrade, N., Paranhos, D., Santos-Neto, E., and Medeiros, R. (2003a). Grid computing for bag of tasks applications. *Proceedings of the Third IFIP Conference on E-Commerce, E-Business and E-Goverment*.

Cirne, W., Paranhos, D., Costa, L., Santos-Neto, E., Brasileiro, F., Sauvé, J., da Silva, F. A. B., Barros, C. O., and Silveira, C. (2003b). Running bag-of-tasks applications on computational grids: The mygrid approach. *In Proceedings of the ICCP'2003 - International Conference on Parallel Processing*, page 407.

Foster, I. (2002). What is the grid? a three point checklist. *GRIDToday*, 1(6).

Foster, I. and Kesselman, C. (1998). The globus project: A status report. *Proc. IPPS/SPDP '98 Heterogeneous Computing Workshop*, pages 4–18.

Foster, I., Kesselman, C., and Tuecke, S. (2001). The anatomy of the Grid: Enabling scalable virtual organizations. *Lecture Notes in Computer Science*, 2150:1.

Medeiros, R., Cirne, W., Brasileiro, F., and Sauvé, J. (2003). Faults in grids: Why are they so bad and what can be done about it? *Proceedings of the 4th International Workshop on Grid Computing*, page 18.