TUXUR - Um Framework para divisão dinâmica de tarefas em Grade Computacional

Roberto Wiest[1], **Benhur de Oliveira Stein**[1]

[1]Programa de Pós-Graduação em Informática (PPGI)
Universidade Federal de Santa Maria (UFSM)

robertow,benhur@inf.ufsm.br

1. Introdução

Grades computacionais são sistemas distribuídos em larga escala que tem como objetivo compartilhar recursos computacionais distintos e dispersos geograficamente. Os recursos são organizados ao redor de *Organizações Virtuais (OVs)* e estes podem fazer parte de uma ou mais OVs de forma coordenada e segura [FOSTER 2001]. Sendo assim, faz-se necessário a utilização de ferramentas capazes de gerenciar essas OVs abstraindo os problemas de segurança de acesso aos recursos.

Algumas das ferramentas atuais como o Ourgrid [CIRNE 2005] e Globus Toolkit [FOSTER 2005] são compostas por um ambiente de execução (middleware) capaz de gerenciar recursos, tarefas e usuários deixando de lado a comunicação e a portabilidade sob responsabilidade do programador. Outra questão que pode tornar difícil a escolha de uma ferramenta é o fato da documentação ser muito extensa, omitindo possíveis limitações.

O presente trabalho visa criar o *framework Tuxur*, esse é capaz de aproveitar os recursos de uma grade de forma simples e eficiente. Dois dos seus principais objetivos são dividir *dinamicamente* uma aplicação conforme a disponibilidade de recursos computacionais disponíveis, bem como explorar novos ambientes de execução.

O Tuxur suporta aplicações maleáveis, isto é, tarefas que sabem se dividir. As tarefas são distribuídas entre os computadores conforme a avaliação das suas capacidades computacionais; as tarefas sofrem divisões de forma a se adaptarem a este parâmetro. *Tuxur* tem a habilidade de se adaptar à capacidade de computação dinamicamente, com adição e remoção de computadores da grade. Além disso, o framework proposto busca explorar novas unidades de processamento, mais especificamente *Graphics Processing Units (GPUs)* com capacidade *General-Purpose computing on Graphics Processing Units (GPGPU)*.

2. A Arquitetura Proposta

A divisão dinâmica é feita adaptando a granularidade das tarefas aos recursos disponíveis na grade. Através de estimativas iniciais sobre a capacidade computacional de um nó ou grupos de nós, uma tarefa é dividida em tarefas menores de forma que estas não sobrecarreguem os nós com menos recursos. Na medida em que as tarefas forem sendo concluídas, a capacidade computacional dos nós é recalculada. A carga em cada nó é mantida em um nível que proporcione tarefa suficiente para evitar que os nós fiquem ociosos. Adicionalmente, existe um mecanismo cujo objetivo é manter os nós que integram a grade ocupados, garantindo que nenhum nó fique ocioso enquanto existirem tarefas a serem computadas.

Projetou-se uma arquitetura para o *Tuxur* a fim de que esse possa comportar novas funcionalidades no futuro, inseridas através de *plugins*. Dentre estas funcionalidades previstas porém ainda não implementadas, podemos citar a interoperabilidade com outras grades computacionais, tolerância a falhas e segurança.

A arquitetura do *Tuxur* é composta de três tipos de módulos, as quais serão descritas a seguir:

- **Módulo Gerente**, principal objetivo do gerente é manter seus trabalhadores ocupados. Para isso, ele sabe qual a carga de cada trabalhador e qual a capacidade de trabalho de cada trabalhador subordinado a ele. O gerente mantém informação do estado de cada uma dos seus trabalhadores e tem que informar seu estado global para seu superior, caso houver.
  Isso permite calcular por quanto tempo aproximadamente ele pode continuar trabalhando e dimensionar quando deve enviar mais trabalho e que quantidade de trabalho deve ser enviada. Como um gerente é visto por ser gerente supervisor como um trabalhador, deve informá-lo sobre o seu estado.
- **Módulo Trabalhador**, é nesse módulo que as tarefas são efetivamente executadas e os resultados computados. Esse contém uma fila com os trabalhos recebidos do gerente. A cada trabalho concluído, esse envia seu resultado ao gerente, envia também informações sobre o seu estado.
- **Módulo Lançador**, esse módulo é responsável por prover um ambiente de comunicação para o gerente e seus demais trabalhadores. Este abre uma porta de comunicação para que as mensagens sejam recebidas, os nós são descritos em um arquivo onde constam informações necessárias para que se possa estabelecer uma conexão inicial via *SSH(Secure Shell)*. Dessa forma, o componente *lançador* lê este arquivo e para cada entrada deste arquivo ele estabelece uma conexão SSH. Criada esta conexão, o *lançador* dispara a execução de um processo passando como argumentos seu endereço IP e o número da porta de comunicação aberta. O processo disparado também abre uma porta de comunicação por onde pode receber novas mensagens do *lançador*. Terminada esta etapa, o processo remoto envia uma mensagem para o *lançador*, informando o seu endereço IP e o número da porta de comunicação aberta. Todas estas informações são armazenadas pelo *lançador*.

Referências

CIRNE, W. (2005). Peer-to-peer grid computing with the ourgrid community. `http://www.ourgrid.org/images/ourgrid/papers/acgc05.pdf`.

FOSTER, I. (2001). The anatomy of the grid. `www.globus.org/alliance/publications/papers/anatomy.pdf`.

FOSTER, I. (2005). A globus primer. `globus.org/toolkit/docs/4.0/key/GT4_Primer_0.6.pdf`.