Resumo

A utilização de grades computacionais para a execução de aplicações paralela é cada vez mais empregada para oferecer um alto poder computacional a um baixo custo. Porém, como os ambientes grades possuem natureza heterogênea, dinâmica, compartilhada e distribuída, adaptar as aplicações para a execução nesse tipo de ambiente de forma a obter um bom desempenho não é uma tarefa simples. Administrar a utilização da rede de comunicação, evitando a ocorrência de congestionamento de mensagens e permitindo uma eficiente utilização da capacidade de processamento e armazenamento oferecida por uma grade, é um desafio ainda maior quando esses recursos cujas características variam são utilizados por pesquisadores que não possuem o conhecimento e nem a habilidade para gerenciar ambientes computacionais complexos.

Uma abordagem já utilizada é adotar um sistema de gerência de aplicações, como por exemplo, o SGA EasyGrid, que esconda do usuário as dificuldades existentes na utilização de uma grade computacional, permitindo que aplicações paralelas possam utilizar as grades computacionais sem grandes esforços. Esse nível de gerenciamento normalmente é capaz de criar aplicações que sejam auto-gerenciáveis, ajustando-se às mudanças que ocorram no ambiente e fornecendo funcionalidades como auto-escalonamento e auto-recuperação de falhas. Entretanto, uma importante atribuição para a gerência da execução de uma aplicação paralela é administrar não só os recursos computacionais, mas também os recursos de comunicação e de armazenamento eficientemente com um mínimo de impacto adverso na execução da aplicação.

Nessa dissertação será estudado o problema de congestionamento da rede de comunicação ocasionado pela troca excessiva de mensagens durante a execução e gerência da aplicação. Será proposta uma nova camada dentro o gerenciador de aplicação *SGA EasyGrid*, permitindo a execução das aplicações autônomas MPI de grande escala sem o risco de congestionamento de mensagens, além de falhar a execução devido a falta de memória para guardar os dados da aplicação.

Abstract

Grid Computing is now extensively being adopted to provide high computational power at low cost for parallel applications. As computational grids are in essence heterogeneous, dynamic, shared and distributed environments, developing grid enabled applications is extremely complex. The difficulty of managing network communication to avoid message congestion and allow the efficient use of the processing power and storage being offered, is even more acute for non-expert users given their lack of familiarity with the intricacies and complexities of computational grids.

A promising approach already being used to address these issues is the creation of autonomic applications through the adoption of Application Management Systems (AMS) that hide from the programmer and user the difficulties related to the grid computing, thus allowing parallel applications to execute efficiently on the grid without much effort. These grid systems generally offer functionalities such as self scheduling and tolerance to resource failures. However, an important aspect of controlling the execution of an application is managing not only processing resources but also communication and storage efficiently with minimal impact on performance. This dissertation focuses on the problem of managing network congestion caused by excessive communication message exchange during the execution of autonomic applications. This work proposes a new layer within the EasyGrid AMS to allow the execution of large scale MPI autonomic applications without the risk of message congestion or even execution failure due to a lack of memory space for the application.