XXX Seminário de INICIAÇÃO CIENTÍFICA DA UFERSA
09 a 12 de dezembro de 2024

Núcleo de Avaliação: Núcleo II
Área temática: Metodologia e técnicas de computação
Área do Conhecimento: Ciência da Computação

Desenvolvimento de Conectores para Comunicação entre Processos Paralelos usando Python e MPI

Ríad Oliveira de Morais, Paulo Henrique Lopes Silva

A computação paralela tornou-se uma abordagem fundamental para solucionar problemas complexos que exigem elevado processamento computacional. Neste contexto, a comunicação eficiente entre processos paralelos é essencial para garantir alta performance na execução de tarefas e coordenação na troca de informações, aspectos críticos em sistemas distribuídos, clusters e supercomputadores. Este trabalho propõe estudar e apresentar as etapas necessárias para o desenvolvimento de uma *Application Programming Interface* (API) intuitiva, capaz de fornecer conectores eficazes para a comunicação entre programas paralelos. Para simplificar a implementação e uso dessa abstração, foi escolhida a linguagem Python em conjunto com a biblioteca *Message Passing Interface* (MPI), através do pacote *mpi4py*, uma adaptação da interface base, que foi originalmente desenvolvida apenas para C e Fortran. A metodologia consistiu inicialmente no estudo de materiais relevantes ao projeto, incluindo trabalhos semelhantes e tópicos da computação paralela, como conceitos, benefícios, limitações e custos associados. Após a fundamentação teórica, foram conduzidas análises e testes da implementação padrão do MPI, por meio do *OpenMPI* em C e C++. Além disso, testes comparativos com o *mpi4py* permitiram verificar diferenças e similaridades em aspectos funcionais e de performance entre as versões. Investigou-se também tecnologias alternativas, como *Remote Procedure Call* (RPC), *Representational State Transfer* (RESTful) APIs, *Inter-Process Communication* (IPC), *ZeroMQ*, *Apache Kafka*, *Sockets*, *Celery*, e *gRPC*. No entanto, optou-se pelo uso dos métodos nativos do MPI para o desenvolvimento dos conectores, especialmente o *Intercomm* e o método Cliente/Servidor, sendo que este último utiliza um servidor intermediário para garantir a comunicação entre processos. Após o desenvolvimento, criou-se um programa para realizar testes de performance no cluster do Núcleo de Processamento de Alto Desempenho (NPAD) da Universidade Federal do Rio Grande do Norte (UFRN). Esse programa, denominado *Parallel Computations*, executa um conjunto de testes simulando um ambiente *Multiple Program Multiple Data* (MPMD), com o objetivo de verificar o custo adicional introduzido pela abstração implementada, chamada *MPMDManager*, em comparação com as versões sem abstração, tanto em Python quanto em C. Os testes foram organizados em ciclos com possibilidade de especificação da quantidade de ciclos desejada. No início de cada ciclo, há comunicação entre os líderes dos programas, que podem ser modificados durante a execução. Em seguida, são realizadas 10 operações computacionais distintas, como multiplicação de matrizes e ordenação de vetores, em ordem fixa. Os resultados indicam que a API desenvolvida

simplifica o código e aprimora a comunicação entre programas paralelos, proporcionando benefícios relevantes, especialmente em Python. Contudo, enquanto as abstrações superam as versões não abstratas em Python, ainda não atingem o desempenho da versão em C. Em conclusão, apesar das limitações de desempenho em relação ao C, a solução oferece uma alternativa eficiente e prática para a comunicação entre processos paralelos.

Palavras-chave: Computação paralela, MPMD, MPI, mpi4py, Intercomm.

Agência financiadora: PICI-UFERSA.

Campus: Mossoró.