Introdução à programação paralela

Gabriel Martins de Miranda – 13/0111350

1. Título do capítulo

Design and Coding of Parallel Programs.

2. Objetivo do capítulo

Aborda os problemas encontrados na construção e codificação de programas paralelos. Apresenta duas abordagens essenciais na construção, que são a **abordagem dirigida a dados paralelos** e a **abordagem dirigida a controle paralelo**. Na primeira, dados são particionados entre processos e cada um executa instruções similares sobre eles. Na segunda, tarefas são particionadas entre os processos, cada um executando comandos diferentes. A maioria dos programas usa as duas abordagens em conjunto, mas a dirigida a dados é mais comum, mais simples e escalável.

3. Resumo do capítulo

Foi vista a construção e codificação de programas paralelos, que podem ser classificados como orientados a dados ou controle. No primeiro, os dados são divididos entre os processos que executam praticamente as mesmas instruções. No segundo, são divididos em tarefas, portanto processos diferentes têm funções diferentes. Há o híbrido, mas o a abordagem a dados é mais comum, já que é mais fácil e escalável.

A orientação a dados foi ilustrada no problema de solução de sistema linear usando método de Jacobi. Foi visto que o método permite fácia paralelização dos dados escalares (n e tolerância de convergência) e vetores, que foram divididos e comunicados entre os processos.

Foi codificado também um programa para ordenar uma lista distribuída, em aboradagem paralela de dados, seguindo o padrão serial de codificação. Foi construído de duas formas, tanto de cima para baixo quanto de baixo para cima, com funções de debug e macros.

Foram vistas duas funções novas, MPI_Alltoall e MPI_Alltoallv, que dividem send_buffer entre processos dentro de comm. No primeiro, o tamanho de cada dado é fixo, enquanto no segundo é configurável.

4. Solução dos exercícios

1. Jacobi paralelo de forma que não assuma que n é divisível por p.

Ok, para contornar este problema, testa-se se n%p é zero, ou seja, n é divisível por p para executar o método. O programa foi rodado e convergiu para os valores de p = 2; n = 2; tolerance = 0.01; iterations = 1000; matriz de entrada = 2 1; 1 2; matriz do lado direito = 3 3; resultando em 1.0 ;1.0. Levou-se em conta que a matriz de entrada deve ser estritamente diagonal dominante, ou seja, o valor absoluto da diagonal na linha deve ser maior que soma absoluta dos outros valores na linha.

2. drand48 e srand48 são usados para gerar doubles aleatórios. Se gerados para processos diferentes, como garantir que serão independentes entre eles?

Para funções de geração de números pseudo-aleatórios serem eficientes é necessário que se tenha um valor semente. Para garantir que a ordem dos números gerados sejam diferentes para cada processo, é necessário que cada um deles tenha uma semente diferente. Isto pode ser feito usando um número que seja exclusivo do processo, tal como seu rank ou múltiplos dele.

5. Programas

1. Use o gerador de números aleatórios para gerar entradas na matriz. Explique o que deve ser feito para as diagonais.

Usou-se o princípio das sementes como sendo o rank dos processos para gerar os números. Para a diagonal, usou-se a soma dos valores da linha mais um valor aleatório.

6. Conclusão

Por meio do capítulo entendeu-se abordagens usadas em programas paralelos, para divisão de dados e controle. Foi possível ter um exemplo prático do método de Jacobi para solução de sistemas lineares. Como o método é bastante paralelizável, foi possível usar a abordagem orientada a dados nele.

7. Referências consultadas

Parallel Programming with MPI by Peter Pacheco ; Design and Coding of Parallel Programs – chapter ten.