Paralelização MPI de métodos iterativos de resolução da equação de Poisson

Computação Paralela

Francisco Resende 84767

Departamento de Física
Universidade de Aveiro
03/07/2020

Conteúdo

Lista de Tabelas

Lista de Figuras

Listings

1 Introdução

De forma a tentar perceber o porquê de existir a necessidade de aplicar paralelização dos métodos iterativos que resolvem a equação de Poisson, é necessário entender o que é a equação de Poisson e como funcionam os métodos iterativos que a resolvem. A equação de Poisson é uma equação diferencial que é comummente utilizada em áreas como a eletrostática, física teórica e engenharia mecânica, a equação de Poisson usada para este trabalho pode ser representada pela equação 1.

$$\frac{d^2V(x,y)}{dx^2} + \frac{d^2V(x,y)}{dy^2} = f(x,y) \tag{1}$$

onde

$$f(x,y) = 2 - x^2 - 10y + 50xy \tag{2}$$

Este tipo de equações de Poisson caracteriza-se, em 2D, como a soma das segundas derivadas parciais espaciais, sendo ela igual a uma função que depende das variáveis espaciais como se pode ver pela equação 1 e pela equação 2.

Dentre os variados métodos iterativos usados para resolução de sistemas lineares, pode-se destacar os métodos de Jacobi e Gauss-Siedel. O método de Jacobi apesar de apresentar uma convergência relativamente lenta, tem um bom funcionamento para sistemas esparsos. No entanto, este método não se comporta bem para todos os casos, para obter um melhor desempenho este método requer que os elementos da diagonal principal sejam não nulos. Mas para que este método seja aplicado à equação diferencial é preciso realizar a sua discretização, para isso foi aplicada uma aproximação por diferenças finitas com um estêncil de 5 pontos, como podemos ver na equação 3 e com um estêncil de 9 pontos como demonstrado na equação 4. Sendo esta aproximação um método de discretização, significa que transforma uma função contínua numa representação discreta, ou seja uma representação ponto a ponto.

$$V^{(k+1)}(i,j) = \frac{1}{4}[V^{(k)}(i-1,j) + V^{(k)}(i,j-1) + V^{(k)}(i,j+1) + V^{(k)}(i+1,j) - h^2 f(i,j)] \tag{3}$$

$$V^{(k+1)}(i,j) = \frac{1}{20}[V^{(k)}(i-1,j-)+4V^{(k)}(i-1,j)+V^{(k)}(i-1,j+1)+4V^{(k)}(i,j-1)+4V^{(k)}(i,j+1)+V^{(k)}(i+1,j-1)+4V^{(k)}(i+1,j) \tag{4}$$

O método de Gauss-Seidel, o segundo método usado neste trabalho, consiste num melhoramento do método de Jacobi que para calcular cada $V^{(k+1)}(i,1)$ usa a solução atual dos quatro vizinhos caso ela já tenha sido calculada, ou seja usa $V^{(k+1)}$ se j<i e $V^{(k+1)}$ se j>i. A expressão para este método pode ser representada pela equação 5

$$V^{(k+1)}(i,j) = \frac{1}{4}[V^{(k+1)}(i-1,j) + V^{(k+1)}(i,j-1) + V^{(k)}(i,j+1) + V^{(k)}(i+1,j) - h^2 f(i,j)] \tag{5}$$

Apesar do método usar valores calculados na iteração atual e outros da iteração anterior, os primeiros valores calculados de cada iteração irão usar valores da iteração passada. Se este último método não for associado a um outro método de auxilio, não irá funcionar quando num programa paralelizado.

Como é possível ver pelas equações 3, 4 e 5 ambos os métodos percorrem todos os pontos do sistema o que faz com que o programa tenha uma complexidade de $n_x * n_y$, onde n_x e n_y são as dimensões do sistema. Como este tipo complexidade computacional aparece a necessidade de dividir o domínio global do sistema em subdomínios e para isso realizar a paralelização do programa. Neste trabalho a paralelização foi totalmente realizada em MPI.