Iniciação a Computação Científica
Trabalho 2

Giancarlo Klemm Camilo
Renan Domingos Merlin Greca

Junho de 2015

Sumário

1 Introdução

O objetivo deste trabalho é a implementação de programa para resolver o PDE:

...

Após o programa inicial foi feito, várias alterações foram feitas para melhorar o desempenho. Os métodos utilizados para análise do código, sistema de testes, otimizações de código e de estruturas de dados são descritas nas seções seguintes.

2 Análise de Arquitetura

3 Limite Superior da Discretização

4 Tempo de Execução

4.1 Programa Original

4.2 Programa Otimizado

5 Análise de Funções

5.1 Programa Original

5.1.1 Método de Gauss-Seidel

5.1.2 Cálculo do Resíduo

5.2 Programa Otimizado

5.2.1 Método de Gauss-Seidel

5.2.2 Cálculo do Resíduo

5.3 Análise dos Dados

6 Otimização do Ponto de Interesse

O ponto de interesse escolhido foi o cálculo do vetor x no método de Gauss-Seidel. Para isso, otimizações foram feitas nas estruturas de dados usadas durante o cálculo e na estrutura do laço em si.

6.1 Estrutura de dados

A estrutura de dados que mais sofreu alterações foi a matriz A. Na versão original do programa, A tinha o tamanho de $((n_x + 1) \times (n_y + 1))^2$, representando a matriz inteira do método analítico de Gauss Seidel.

Olhando para a matriz A, percebemos que grande parte das posições tinham valor 0 e que os valores de interesse de cada linha estavam numa distância de $(n_y + 1)$ da diagonal principal da matriz. Ou seja, os dados que estavam além desse intervalo eram sempre 0 e poderiam ser ignorados.

Além disso, percebemos que as posições ao redor da diagonal principal sempre seguiam o seguinte padrão:

$$h_x \ \texttt{0} \ h_y \ \texttt{1} \ h_y \ \texttt{0} \ h_x$$

Onde o elemento na diagonal principal é sempre 1, h_x e h_y representam a dependência dos pontos adjacentes e, no exemplo acima, $n_y = 2$. Sendo assim, podíamos ignorar as posições que sempre continham 1 ou 0, além de evitar a repetição de h_x e h_y.

Também foi possível ver que os valores de h_x e h_y permaneciam constantes em quase todas as linhas da matriz, exceto nas linhas em que não estavam presentes. As linhas que não continham h_x e h_y representavam os pontos das bordas da grade, que são calculadas separadamente. Logo, foi possível ver que uma matriz que simplesmente nos dizia se um determinado ponto é ou não uma borda era suficiente para fazer os cálculos de Gauss-Seidel, se salvássemos h_x e h_y em variáveis separadas.

Portanto, na versão atual, a matriz A tem o tamanho de $(n_x+1)\times(n_y+1)$ e utiliza o tipo de dados *short int*, pois apenas armazenamos 0 quando o ponto é uma borda ou 1 caso contrário.

6.2 Código

Na versão anterior do programa, o laço de Gauss-Seidel continha quatro desvios condicionais, um para cada borda. Nesta versão, temos apenas um

desvio que utiliza a matriz A para verificar se o ponto em questão é borda ou não.

Caso o ponto não seja uma borda, quatro operações são realizadas utilizando os valores de h_x e h_y e outras posições do vetor x. Na versão original do programa, as quatro operações eram armazenadas na variável `temp` utilizando o operador `+=`.

7 Resultados

7.1 Tempo

7.2 Memória