CSC 64 Exercício 04

CE 265 Exercício 04

1. Finalidade

Vetorizar laços e programas.

2. Preparo no SDumont

Cada vez que entrar no SDumont, digite os comandos *module load python* e *module load intel_psxe*, nessa ordem. Esses comandos instalam o interpretador *python* e o compilador *intel icc* versão 19.1.3.304, que utilizaremos neste exercício.

3. Relatório de Vetorização

Compiladores vetorizadores geram relatório de vetorização. Na versão do compilador *icc* instalada pelo comando *module load intel_psxe*, a geração do relatório é acionada pela chave de compilação *-qopt-report-phase=vec*, inclusa nos *Makefile* fornecidos. Ao compilar o arquivo *X.c* o compilador gera o relatório de vetorização no arquivo *X.optrpt*.

Relatórios de vetorização de um arquivo são divididos por procedimento. Laços de um procedimento são identificados pelo número da linha (por exemplo, "LOOP BEGIN at Lacos.c(35,3)" refere-se ao laço iniciado na linha 35 do arquivo Lacos.c Cada laço é classificado como vetorizado ("LOOP WAS VECTORIZED"), parcialmente vetorizado ("PARTIAL LOOP WAS VECTORIZED") ou não vetorizado ("loop was not vectorized"). O relatório prossegue com motivos para não vetorizar um laço, tanto na mesma linha quanto nas linhas posteriores.

4. Arquivo com os Programas deste Exercício

Encontre nesta atividade o arquivo *Exercicio04.tgz*. Leve-o para sua área no SDumont. Como sempre, desempacote o arquivo no diretório $SCRATCH. A operação de desempacotar produz o diretório *Exercicio04* que contém dois outros diretórios, *Lacos* e *Calor*.

5. Laços

Esse diretório contém o programa fonte *Lacos.c*, *Makefile* e o script *Xmit.py*.

O arquivo *Lacos.c* contém duas funções a vetorizar, duas funções auxiliares e programa principal.

As funções auxiliares são *Init*, que inicializa vetores e *Soma*, que soma todos os elementos do vetor de entrada.

As funções a vetorizar são *Laco1* e *Laco2*.

O programa principal inicializa os vetores, imprime as somas dos vetores relevantes para *Laco1*, invoca a função *Laco1*, imprime as somas dos mesmos vetores, inicializa novamente os vetores, imprime as somas dos vetores relevantes para *Laco2*, invoca *Laco2* e imprime as somas dos mesmos vetores.

Após carregar os dois módulos, compile utilizando o *Makefile*. Observe que o relatório de vetorização reporta que o laço na linha 35, procedimento *Laco2* não foi vetorizado. O relatório também informa que o laço na linha 25, procedimento *Laco1*, foi parcialmente vetorizado.

Execute invocando *Xmit.py* sem argumentos. Resulta o arquivo *Out_Lacos.txt*, que contém a soma dos vetores relevantes antes e depois das invocações das funções *Laco1* e *Laco2*. Reserve o arquivo produzido.

Sua tarefa é alterar o corpo das funções *Laco1* e *Laco2* de forma a vetorizar totalmente seus laços. Ignore *Soma*, *Init e main*, mantendo-os inalterados. Após vetorizar, execute novamente *Xmit.py* e guarde *Out_Lacos.txt*.

São corretas as vetorizações que produzem o mesmo resultado da versão fornecida, ou seja, que o arquivo *Out_Lacos.txt* produzido pela versão vetorizada seja idêntico ao produzido pela versão original.

Observe que Out_*Lacos.txt* retrata apenas os argumentos de entrada e de saída das funções. As variáveis locais das funções são irrelevantes. Ou seja, ao vetorizar as funções, sinta-se livre para eliminar, inserir ou modificar as variáveis locais das funções *Laco1* e *Laco2*, contanto que a assinatura das funções (os argumentos de entrada e saída) mantenham-se inalteradas.

6. Calor

O arquivo *Calor.c* contém a solução estacionária da Equação de Calor ($\nabla^2 B_{x,y} = 0$) em uma chapa quadrada com uma borda permanentemente a 10 graus e as demais bordas permanentemente a 0 graus, onde $B_{x,y}$ representa a temperatura em cada ponto da chapa. A chapa é discretizada por uma grade bidimensional $(n+2) \times (n+2)$ e as temperaturas são armazenadas no *array* $b[n+2][n+2]$. A primeira e a última linha e coluna do *array* representam as bordas da chapa, com temperaturas mantidas constantes ao longo da computação.

O arquivo contém cinco funções e um programa principal. Três funções são auxiliares: *Init* inicializa uma borda da grade para dez graus e o restante da grade para zero graus; *Dump* imprime o interior da grade para eventual "debug" e *MaxDif* calcula a maior diferença relativa entre duas grades.

As duas outras funções, denominadas *CalorEscalar* e *CalorVetorial* são idênticas. Resolvem o sistema de equações lineares resultante da discretização da equação de calor por diferenças finitas utilizando um método iterativo (Gauss-Siedel) com *nIter* iterações.

O programa principal declara dois valores para n (500 e 10.000), com o maior deles comentado. Segue inicializando a grade e resolvendo a equação duas vezes, a primeira com *CalorEscalar* e a segunda com *CalorVetorial*. Termina calculando e imprimindo a maior diferença relativa entre as duas soluções e o tempo de execução de cada solução. Invocações de *Dump* estão comentadas.

Após carregar os dois módulos, compile utilizando o *Makefile* e execute utilizando *Xmit.py* sem argumentos. Produzirá o arquivo de saída *Out_Calor.txt*.

O relatório de vetorização mostra nenhum laço vetorizado em *CalorEscalar* e *CalorVetorial*.

Sua tarefa é vetorizar *CalorVetorial*.

Não é necessário vetorizar essa função mantendo a estrutura de dados da grade. Sinta-se à vontade para utilizar outras estruturas de dados que considere mais adequadas para a vetorização, desde que locais à função. Ou seja, a assinatura da função (argumentos de entrada e de saída) deve permanecer inalterada. O uso de memória em excesso é irrelevante.

Para vetorizar, sugiro trabalhar na grade com o menor valor de n. Após vetorizar, execute *Xmit.py* para o maior valor de n, reservando o arquivo produzido. Mantenha a execução de *CalorEscalar*, permitindo comparar os tempos de execução das duas versões e a diferença nos resultados numéricos.

Funções vetorizadas cuidadosamente codificadas produzem resultados numéricos idênticos aos da função escalar.

7. Relatório

Monte arquivo .tgz com os programas vetorizados e entregue em conjunto com seu relatório.

Componha relatório em pdf com o seguinte conteúdo:

1. Explique e justifique como vetorizou *Laco1* e *Laco2*.
2. As funções *Laco1* e *Laco2* vetorizadas *("cut and paste")*.
3. O relatório de vetorização dessas funções. Bastam as linhas informando os laços vetorizados.
4. Explique e justifique como vetorizou *CalorVetorial*.
5. A função *CalorVetorial* vetorizada *("cut and paste")*.
6. O relatório de vetorização de *CalorVetorial*. Novamente, bastam as linhas informando os laços vetorizados.
7. O arquivo de saída da execução com o maior valor de n.
8. Calcule o ganho ("*speed-up*") obtido na vetorização de *Calor* com o maior valor de n. Caso você conseguisse paralelismo OpenMP perfeito, quantas threads seriam necessárias para obter ganho semelhante ao da vetorização?
9. Seu comentário sobre o exercício.

Retorne o relatório e o arquivo .tgz até a meia noite imediatamente anterior à próxima aula.