Universidade Estadual de Campinas – UNICAMP
Faculdade de Tecnologia – FT

T_TT005B_2020S2 - Tópicos Especiais em Telecomunicações I

(Programação de Alto Desempenho)

Grupo: *Gáta moustáki*
Paloma Eduarda Salvador de Mello - RA 260610
Henrique Arakaki Fonseca - RA 236540

1. Resolução

Utilizou-se a biblioteca OpenACC para usufruir do processamento da GPU com intuito de comparar os tempos de execução entre a CPU e a GPU. Para isso, fez-se uso da instância virtual da Amazon Web Services (o MobaXterm foi usado para se conectar a essa instância) equipada com uma placa gráfica NVIDIA Tesla k80. Para facilitar sua compilação e evitar erros, sua instrução de compilação (*gcc -fopenacc prog.c -o prog*) foi salvo no arquivo "compilador.sh", que encontra-se salvo no repositório GitHub criado pelo grupo (https://github.com/Shokitex/Labs-High-Performance-Programming) juntamente com o arquivo prog.c, que contém o código fonte e uma cópia deste PDF.

Para compilar e executar o programa, deve-se baixar os arquivos compilador.sh e prog.c para o computador. No prompt de comando, após selecionar a pasta onde os arquivos foram salvos deve-se digitar a seguintes linhas de comando:

chmod +x compilador.sh

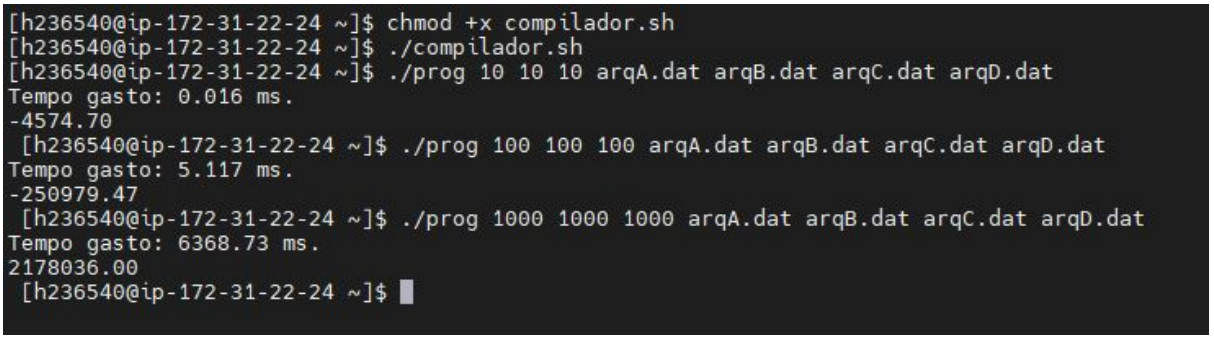
./compilador.sh

Em seguida pode-se executar o programa seguindo a sintaxe:

./lab02 y w v arqA.dat arqB.dat arqC.dat arqD.dat

Na imagem abaixo, é possível observar o processo de compilação e execução do programa, assim como explicado anteriormente, bem como suas saídas.
Fez-se os seguintes testes:

y = 10, w = 10 e v = 10
y = 100, w =100 e v =100
y = 1000, w = 1000 e v = 1000



Na saída do programa tem-se a redução da matriz D, gerada por (matriz A x matriz B) x Matriz C, e os tempos de execução em ms (milissegundos) para cada valor de y, w e v, descontadas as operações de I/O.

Copiou-se para a GPU as matrizes A,B,C,D e matriz auxiliar - alocadas dinamicamente na CPU - buscando um melhor desempenho nos cálculos (entende-se por desempenho um menor tempo de execução). Após feito os cálculos, trouxemos da GPU apenas a matriz que nos interessa, a matriz D, enquanto as outras matrizes foram excluídas de dentro da unidade.

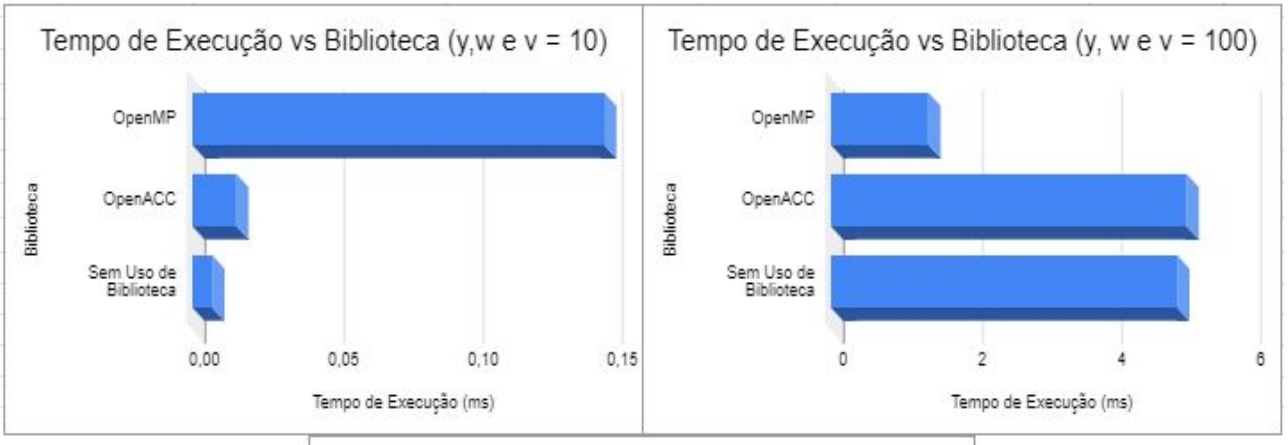
2. Comparações

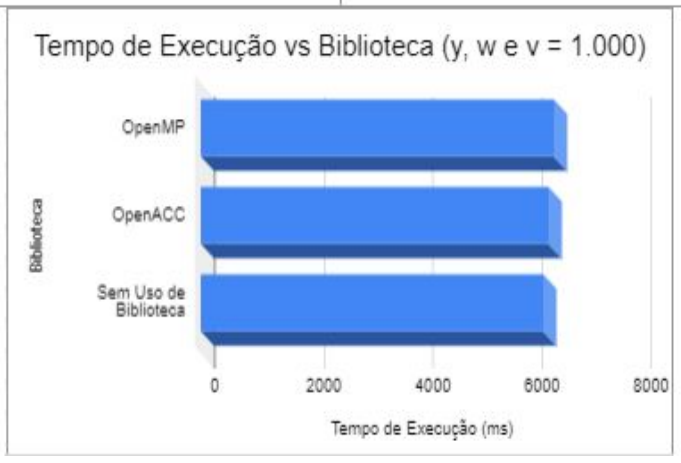
Abaixo temos a comparação entre os tempos de execução do programa utilizando a OpenMP com 4 núcleos de processamento (CPU) e a OpenACC(GPU). Para fins de estudo, comparou-se também com a execução do programa sem uso de nenhuma biblioteca.

Tabela relacionando biblioteca, valores de entrada e o tempo de execução:

Biblioteca	Entradas	Tempo de Execução (ms)
OpenMP	y = 10, w = 10 e v = 10	0,148
	y = 100, w =100 e v =100	1,395
	v = 1000, w = 1000 e v = 1000	6463,31
OpenACC	y = 10, w = 10 e v = 10	0,016
	y = 100, w =100 e v =100	5,117
	v = 1000, w = 1000 e v = 1000	6368,73
Sem Uso de Biblioteca	y = 10, w = 10 e v = 10	0,007
	y = 100, w =100 e v =100	4,97
	v = 1000, w = 1000 e v = 1000	6267,98

Gráficos gerados para cada valor de y, w e v:





3. Conclusões

Após a realização dos testes com a OpenMP e OpenACC, percebe-se que ao realizar contas menores, como multiplicações de matrizes 10x10 e 10x1, o programa sem o uso das bibliotecas se torna mais eficiente em comparação ao que usa a biblioteca OpenACC, pois no gráfico y,w e v = 100 há o tempo extra do carregamento das matrizes necessárias para dentro da GPU e a devolução da matriz resultante para a memória na CPU. Em relação a biblioteca OpenMP, percebe-se que em programas de médio porte ela se torna mais eficiente, como visto no segundo gráfico (y, w e v = 100), pois as threads da CPU são mais rápidas em relação as threads da GPU. Porém quando temos programas de pequeno ou grande porte (poucos cálculos ou muitos cálculos) , sua eficiência é reduzida, pois no primeiro caso, a criação de suas threads acrescenta um tempo considerável na execução do programa, já no segundo, por ter um número limitado de threads em comparação a biblioteca OpenACC, os cálculos acabam ocupando de espaço maior de tempo.

Assim, conclui-se que é importante levar em consideração a quantidade de operações que se espera realizar com o programa para uma percepção melhor de qual biblioteca usar. A OpenACC, por possuir grande quantidade de threads, é uma boa opção quando temos grande quantidade de cálculos que podem ser realizados paralelamente, tornando a execução mais rápida. Mas deve-se tomar cuidado, pois o tempo de carregamento de dados da CPU para a GPU e vice-versa é consideravelmente lento.