41145

Page No.

Date 29 9 21

LP 1 - HPC 2

* Title : Vector and Matrix Operations

* Problem Statement :
Design parallel algorithms to -
i) Add 2 large vectors
ii) Multiply 2 N x N arrays using n^2 processors.
iii) Multiply vector and a matrix.

* Objectives:
Learn parallel programming using OpenMP.

* Outcomes:
i) Design parallel algorithm.
ii) Use OpenMP for task parallelism.

* Pre-requisites:
Multithreading and basics of OpenMP

* Theory:
OpenMP is an API that can be used with languages like C++, C, FORTRAN for programming shared address space machines. OpenMP directives provide support for concurrency, synchronization and data handling while obviating the need for explicitly setting up condition variables, data scope and initialization.

OpenMP programs execute serially until they encounter the parallel directive. This directive is responsible for creating a group of threads.

41145

Page No.
Date

The exact no. of threads can be specified in the directive.
The main thread that encounters the parallel directive before the master of this group of threads and is assigned a thread id 0 within the group.
The parallel directives have the following prototype-

```
#pragma exp parallel [ clause list]
```

* Matrix vector product:
To define the multiplication bet^n a matrix and a vector, we need to consider the vector as a column matrix.
We define the matrix-vector product only for the case when the no. of rows in the vector is equal to the no. of column's of the matrix.

$$Ax = \underset{(m\times n)}{\begin{bmatrix} a_{11} & a_{12} & a_{13} \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots\cdots & a_{2n} \\ \vdots & & & \\ a_{n1} & \cdots & \cdots & a_{mn} \end{bmatrix}} \underset{(n\times 1)}{\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}}$$

$$= \begin{bmatrix} a_{11}\cdot x_1 + a_{12}\cdot x_2 + \cdots \\ a_{21}\cdot x_1 + a_{22}\cdot x_2 + \cdots\cdots \\ \vdots \\ a_{m1}\, x_1 + a_{m2}\cdot x_2 + \cdots a_{mn}\, x_n \end{bmatrix}$$

* Algorithm: (Matrix - vector mult.)

① Read the size of the matrix (M x N) and read the elements of the matrix randomly.
② Read the size of the vector (N) and its elements.
③ Read start time
④ Using #pragma exp parallel for parallel multiplication of the matrix and vector, row by row.
⑤ Read the end time
⑥ Calculate and display the execution time as (end time - start time).
⑦ Repeat this procedure for various matrix sizes.

Algorithm: (Matrix - matrix multiplication)

① Read the size of the matrices (N x N) and read their numbers randomly.
② Read the start time.
③ Using #pragma emp parallel for parallel multi row by row.
④ Read the end time.
⑤ Calculate and display the execution time as (end time - start time)
⑥ Repeat this procedure for various matrix sizes.

41145

Page No.
Date

Algorithm - (Vector Addition)

① Read the size of the vector (N) and read the numbers.
② Read the start time.
③ Using #pragma omp parallel for ~~mode~~ parallel execution of single for loop and perform the vector addition.
④ Read the end time.
⑤ Calculate and display the execution time as (end time - start time)
⑥ Apply this for various vector sizes.

* Conclusion:

From this assignment, I was able to understand the basics of Open MP and parallel programming and hence implement this assignment.