PAGE NO.:
DATE : / /

Assignment No:

Problem Statement: Vector and matrix operations.
Design Parallel algorithm to:
i) Add two large vectors.
ii) Multiply vector and matrix.
iii) Multiply two N×N arrays using n^2 processors.

Objectives:
i) To understand vector and matrix operations
ii) To implement parallel algorithm to perform matrix and vector operations.

Outcomes:
i) We will implement vector and matrix operations using parallel algorithms

Requirements:
64 bit OS Linux
Google Colab

Theory:

While executing the parallel algorithms of matrix vector multiplication it is necessary to distribute not only the matrix A, but also the vector b, and the result vector C.

If the processor holds the matrix row and all the elements of the vector b to the c the total number of used memory is the same order O(n).

Pratibha

PAGE NO.:

DATE : / /

Matrices and matrix operations are widely used in mathematical modeling of various processors, phenomenon and systems. Matrix based on many scientific and engineering calculations, computational mathematics physics, economics are only some of the areas of their applications.

The efficiency of carry out matrix computation is highly important many standard libraries contain procedure for various matrix multiplications.

Add two large vectors:
When added together in this different order these same three vectors still produce a resultant with the same magnitude and direction as before. The order in which vectors are added using the head to tail methods insignificant.

Vector implements a dynamic array. It is similar to ArrayList but with two differences, vector is synchronized vectors contain many legacy methods that are not part of collective framework.

Two add or substract 2 vectors in corresponding vectors components, Let $u \rightarrow \langle u_1, u_2 \rangle$ and $v \rightarrow \langle v_1, v_2 \rangle$ be two vectors. The sum of two or more vectors is called the resultant. The resultant

Pratibha

PAGE NO.:
DATE: / /

of two vectors can be found using either the using parallel algorithms or parallelgram method for the triangle method.

E.g.

$$\begin{bmatrix} A_x & A_y & A_z \end{bmatrix} \begin{bmatrix} B_x \\ B_y \\ B_z \end{bmatrix} = A_x B_x + A_y B_y + A_z B_z \Rightarrow \vec{A} \cdot \vec{B}$$

Multiplying vector and matrix:
This is the same as standard matrix multiplications. Lets multiply the rows of the matrix by the column of the vector which is the same as in regular matrix multiplication when two multiply when the rows of the first matrix by the columns of second matrix.

A matrix is simply a rectangular array of numbers and vectors are row of the matrix. A vector can be considered as 1 by n matrix or n by 1 matrix. the basic usefulness of matrices is to represent line or transformation of vectors or linear mappings between vector spaces.

E.g.

$$X = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} * \begin{bmatrix} 9 \\ 8 \\ 7 \end{bmatrix}$$

$$= \begin{bmatrix} 1\times9 + 2\times8 + 3\times7 \\ 4\times9 + 5\times8 + 6\times7 \end{bmatrix} = \begin{bmatrix} 46 \\ 118 \end{bmatrix}$$

Prabhat

DATE : / /

CUDA Programming:

Let M and N be the input vectors (or matrices) and P be the result obtained from M and N. then,

P = M+N (Vector addition)
P = M×N (Vector-Matrix multiplication)
P = M*N (Matrix multiplication)

Here each element in P can be obtained from 1 thread.
Thus the problem is decomposed into n threads for # parallel conditions using thread IDs for differentiating data.

Test Cases and Analysis:

	Operation	Input Size	Sequential Time	Parallel Time	Efficiency
1.	Vector Addition	n=256	0.01	0.02	0.5
		n=1024	0.01	0.02	05
		n=2048	0.02	0.01	2.0
2.	Vector-matrix multiplication	n=256	0.003	0.082	0.031
		n=1024	0.093	0.135	0.689
		n=2048	0.367	0.133	2.759

PAGE NO.:
DATE : / /

3. Matrix multiplication	n=256	0.480	0.02	24.0
	n=1024	0.620	0.133	4.66
	n=2048	0.754	0.136	5.544

Efficiency: $\frac{WCSA}{WCPA}$

- Here we observe that parallel algorithm is a major improvement for matrix-matrix multiplication while its successfully significant for increasing input size for vector-matrix multiplication.

- No major improvement is gained for vector additions of similar size, but it gets better for large value of n.

Input:
Vector 1: [5 7 9 11 4 7 6 2 0 1]
Vector 2: [6 2 1 0 4 13 17 12 15 2]

Output:
Vector 3: [11 9 10 11 8 20 21 14 15 3]

Conclusion:

Thus we implemented vector addition, matrix-vector multiplication, matrix-matrix multiplication problems using sequential and parallel computings.

Pratibha