Assignment No : 1 (A)

- Title : Design & implement parallel Breadth First Search based on existing algorithms using OpenMP. Use a tree or an undirected graph for BFS.

- Objectives : Students should be able to perform Parallel BFS based on existing algorithms using OpenMP.

- Prerequisite : 1. Basic of programming language
  2. Concept of BFS
  3. Concept of parallelism.

- Theory :

What is BFS ?

BFS stands for Breadth-First Search. It is a graph traversal algorithm used to explore all the nodes of a graph or tree systematically, starting from the root node or a specified starting point, and visiting all the neighboring nodes at the current depth level before moving on to the next depth level.

- The algorithm uses a queue data structure to keep track of the nodes that need to be visited & marks each visited node to avoid processing it again.
- Used for solving puzzles.

- Example of BFS :

Step 1 : Take an Empty queue
Step 2 : Select a starting node & insert it into queue
Step 3 : Provided that the queue is not empty, extract the node from queue & insert its child nodes.
Step 4 : Print the extracted node.

CLASSMATE
Date :
Page :

- How parallel BFS Work ?
  - It is used to explore all the nodes of graph or tree systematically in parallel. It is popular parallel algorithm used for graph traversal in distributed computing, shared-memory systems, and parallel clusters.
  - It uses two phases : the computation phase & the communication phase.

- Conclusion : In this way we can achieve parallelism while implementing BFS.

Assignment No : 01 (B)

- Title : Design and implement Parallel DFS based on existing algorithm using OpenMP. Use a tree or an undirected graph for DFS.

- Objective : Students should be able to perform parallel DFS based on existing algorithms using OpenMp.

- Prerequisite : 1. Basic of programming language.
  2. Concept of DFS
  3. Concept of Paralledism.

- Theory :

What is DFS ?

- DFS stands for Depth-First Search. It is popular graph traversal algorithms that explores as far as possible along each branch before backtracking. This algorithm can be used to find the shortest path between two vertices or to traverse a graph in a systematic way.
- DFS can be implemented using either a recursive or an iterative approach.

- Concept of OpenMP :

- OpenMP (Open Multi Processing) is an application prog. Interface (API) that supports shared-memory parallel programming in C, C++ and Fortran.
- It is used to write parallel programs that can run on multicore processors, multiprocessor systems and parallel computing clusters.

How parallel DFS work ?

- Parallel DFS is an algorithm that explores the depth of a graph structure to search for nodes.
- Parallel DFS works by dividing the graph into smaller subgraphs that are explored simultaneously.
- Each processor or thread is assigned a subgraph to explore, and they work independently to explore the subgraph using standard DFS algorithm.
- Parallel DFS can be implemented using several parallel programming models such as OpenMP, MPI and CUDA.

- Conclusion : In this way we can achieve parallelism while implementing DFS.

Assignment No : 02 (A)

- Title : Write a program to implement parallel bubble sort.

- Objectives : Students should be able to write a program to implement parallel bubble sort and can measure the performance of sequential and parallel algorithms.

- Prerequisite : 1. Basic of programming language.
  2. Concept of bubble sort.
  3. Concept of parallelism.

- Theory :

What is bubble sort ?

- It is a simple sorting algorithm that works by repeatedly swapping adjacent elements if they are in the wrong order. It is called "bubble" sort because the algorithm moves the larger elements towards the end of the array in a manner the resembles the rising bubbles in a liquid.

Algorithm :

1. Start at the beginning of the array.
2. Compare the first 2 elements. If the first element is greater than second element, swap them.
3. Move to the next pair of element & repeat step 2.
4. Continue the process until the end of the array is reached.
5. If any swaps were made in step 2-4, repeat the process from step 1.

Time complexity : $O(n^2)$

How parallel bubble sort work ?

- In parallel bubble sort, the list of elements is divided into multiple sublists that are sorted concurrently by multiple threads.
- Each thread sorts its sublist using the regular bubble sort algorithm. When all sublists have been sorted, they are merged together to form the final sorted list.
- The parallelization of the algorithm is achieved using OpenMP.
- In the parallel bubble sort algorithm, the main loop that iterates over the list of elements is divided into multiple iterations that are executed concurrently by multiple threads.

- Conclusion : In this way we can implement bubble sort in parallel way using OpenMP.

Assignment No : 02 (B)

- Title : Write a program to implement parallel merge sort.
- Objective : Students should be able to write a program to implement parallel merge sort.
- Prerequisite : 1. Basic of programming language.
  2. Concept of merge sort.
  3. Concept of parallelism.
- Theory :

What is merge sort ?

- Merge sort is a sorting algorithm that uses a divide and conquer approach to sort an array or a list of elements.
- The algorithm works by recursively dividing the input array into two halves, sorting each half and then merging the sorted halves to produce a sorted output.

Steps :

1. Divide the input array into two halves.
2. Recursively sort the left half of the array.
3. Recursively sort the right half of the array.
4. Merge the two sorted halves into a single sorted output array.

How parallel merge sort works :

- It is a parallelized version of the merge sort algo. that takes adv. of multiple processors or cores to

improve its performance. In this the input array is divided into smaller subarrays, which are sorted in parallel using multiple processors or cores.

Steps :

1. Divide the input array into smaller subarrays.
2. Assign each subarray to separate processor or core for sorting.
3. Sort each subarray in parallel using the merge sort algorithm.
4. Merge the sorted subarrays together in parallel to produce the final sorted output.

- Conclusion : In this way we can implement merge sort in parallel way using OpenMP.

Assignment No : 03

- Title : Implement Min, Max, Sum and Average operations using parallel reduction.
- Objective : To understand the concept of parallel reduction and how it can be used to perform basic mathematical operations on given datasets.
- Prerequisite : 1. Parallel programming architectures.
  2. Parallel programming models.
  3. Proficiency in programming languages.
- Theory :

Parallel Reduction :

1. Min_Reduction function :
   - The function takes in a vector of integers as input and finds the min. value in the vector using parallel reduction.
   - The OpenMP reduction clause is used with the "min" operator to find the minimum value across all threads.
   - The minimum value found by each thread is reduced to the overall minimum value of the entire array.
   - The final minimum value is printed to the console.

2. Max_Reduction function :
   Same as Min_Reduction function.

3. Sum_Reduction function :
   - The function takes in a vector of integers as input and finds the sum of all the values in the vector using parallel reduction.

- The OpenMP reduction clause is used with the "+" operator to find the sum across all threads.
- The sum found by each thread is reduced to the overall sum of the entire array.
- The final sum is printed to the console.

4. Arerage_Reduction function:
- The function takes in a vector of integers as input & finds the average of all the values in the vector using parallel reduction.
- The OpenMP reduction clause is used with the "+" operator to find the sum across all threads.
- The sum found by each thread is reduced to the overall sum of the entire array.
- The final sum is divided by the size of the array to find the average.

5. Main function:
- The function initializes a vector of integers with some values.
- The function calls the min_reduction, max_reduction, sum_reduction and avg_reduction functions on the input vector to find the corresponding values.

6. Compiling & running the program:
1. Compile the program.
2. Run the program.

- Conclusion: In this way we can implement these operations using parallel reduction.

Assignment No : 04 (A)

- Title : Write a CUDA program for addition of two large vectors.
- Objectives : Students should be able to perform CUDA program for addition of two large vectors.
- Prerequisite : 1. CUDA concept
  2. Vector addition
  3. How to execute program in CUDA.
- Theory :

What is CUDA ?

- CUDA (Compute Unified Device Architecture) is a parallel computing platform and programming model developed by NVIDIA.
- It allows developers to use the power of NVIDIA GPUs to accelerate computation tasks in various app, including scientific computing, ML.
- It provides a set of programming APIs, libraries & tools that enable developers to write & execute parallel code on NVIDIA GPUs.

Steps for Addition of two large vectors using CUDA :

1. Define the size of the vectors :
2. Allocate memory on the host :
3. Initialize the vectors :
4. Allocate memory on the device :
5. Copy the input vectors from host to device :
6. Launch the Kernel :
7. Copy the result vector from device to host :

8. Free memory on the device :
9. Free memory on the host :

Execution of program :
1. Install CUDA toolkit
2. Setup CUDA environment
3. Write the CUDA program
4. Compile the CUDA program
5. This will generate an executable program named program_name.
   Run the CUDA program

- Conclusion : In this way we can write the CUDA program for addition of two large integers

Assignment No : 04 (B)

- Title : Write a program for matrix multiplication using CUDA C.
- Objectives : Students should be able to perform program for matrix multiplication using CUDA C.
- Prerequisite : CUDA concept
  2. Matrix multiplication
  3. How to execute program in CUDA.
- Theory :

Steps for implementing matrix multiplication :

1. Matrix initialization
2. Memory allocation.
3. Data transfer.
4. Kernel launch
5. Device synchronization
6. Data retrieval
7. Memory Deallocation.

Execution of program over CUDA environment :

1. Install CUDA Toolkit
2. Set up CUDA environment
3. Write the CUDA program
4. Compile the CUDA program
   nvcc -o program_name program_name.cu
5. Run the program
   ./ program_name.

Conclusion : In this way we can write the program for matrix multiplication using CUDA.

Assignment No : 01

- Title : Linear regression by using Deep Neural Network : Implement Boston housing price prediction problem by linear regression using Deep Neural Network. Use Boston House price prediction dataset.

- Objectives : Students should be able to perform linear regression by using Deep Neural network on Boston House Dataset.

- Prerequisite : 1. Basic of programming language.
  2. Concept of Linear regression.
  3. Concept of Deep Neural Network.

- Theory :

What is Linear Regression ?

- It is a statistical approach that is commonly used to model the relationship between a dependent variable and one or more independent variable.
- It assumes a linear relationship between the variables and uses mathematical methods to estimate coefficients that best fit the data.
- Linear regression using deep neural networks combines the principles of linear regression with the power of deep learning algorithms.
- Example : Predicting the price of house based on various features such as size of house, no. of bedrooms, the location etc.

Cocept of Deep Neural Network :

- A deep Neural Network is a type of ML algorithm that is modeled after the structure and function of the humain brain.

- It consists of multiple layers of interconnected nodes or artificial neurons, that process data and learn from it to make predictions or classifications.
- Each layer of the network performs a specific type of processing on the data, such as identifying patterns or correlations between features and passes the results to the next layer.
- The layers closest to the input are known as "input layer" while the layers closest to the output are known as "output layer".
- The intermediate layers between the input & output layers are known as 'hidden layers". These layers are responsible for extracting increasingly complex features from the input data, and can be deep or shallow.

How Deep Neural Network works ?

1. Data preprocessing.
2. Model architecture
3. Model training
4. Model evaluation
5. Model prediction.

The dataset includes 13 input features, CRIM, ZN, INDUS, CHAS, NOX, RM, AGE etc.

- Conclusion : In this way we can predict the Boston House price using Deep Neural Network.

Assignment No : 02 (A)

- Title : Binary classification using Deep Neural Network Example : Classify movie reviews into positive" reviews and "negative' reviews, just based on the text content of the reviews. Use IMDB dataset.

- Objective : Students should be able to classify movie reviews into positive and negative reviews.

- Prerequisite : 1. Basic of programming language.
  2. Concept of classification.
  3. Concept of deep neural network.

- Theory :

What is classification ?

- Classification is a type of supervised learning in ML that involves categorizing data into predefined classes or categories based on a set of features or characteristics.
- It is used to predict the class of new, unseen data based on the patterns learned from the labeled training data.
- In classification, a model is trained on a labeled data-set, where each data point has a known class label.
- The model learns to associate the input features with the corresponding class labels and can then be used to classify new, unseen data.

How Deep Neural Network Work on Classification :

- Deep neural networks are commonly used for classification tasks because they can automatically learn to extract relevant features from raw input data & map them

to the correct output class.
- The basic architecture of a deep neural network for classification consists of 3 main parts: an input layer, one or more hidden layer and an output layer.
- During training, the deep neural network learns to adjust its weight and biases in each layer to minimize the difference between the predicted output and true labels.
- This is typically done by optimizing a loss function that measures the discrepancy between the predicted and true labels, using techniques such as gradient desent or stochastic gradient desent.

- IMDB Dataset : The IMDB dataset is a large collection of movie reviews collected from the IMDB website, which is a popular source of user-generated movie ratings and reviews. The dataset consists of 50,000 movie reviews, split into 25,000 reviews for the training and 25,000 reviews for testing.

- Conclusion : In this way we can classify the movie reviews by using DNN.

Assignment No : 02 (B)

- Title : Multiclass classification using Deep Neural network : Example : Use the OCR letter recognition dataset.

- Objectives : Students should be able to solve Multiclass classification using Deep Neural Networks.

- Prerequisite : 1. Basic of programming language.
  2. Concept of Multi-classification
  3. Concept of Deep Neural Network.

- Theory :

What is Multiclass classification ?

- Multi classification, also known as multiclass classification or multiclass classification problem, is a type of classification problem where the goal is to assign input data to one of three or more classes or categories.
- In other words, instead of binary classification, where the goal is to assign input data to one of two classes, multiclass classification involves assigning input data to one of several possible classes or categories.
- In multiclass classification, each input sample is associated with a single class label and the goal of the model is to learn a function that can accurately predict the correct class label for new, unseen input data.
- It can be approached using a variety of ML algorithms, including decision trees, support vector machines and deep neural networks.

- Example :
1. Image classification
2. Text classification.
3. Disease diagnosis.
4. Speech recognition
5. Credit risk analysis.

- Conclusion : In this way we can do Multiclassification using DNN.

Assignment No : 03

- Title : Use MNIST Fashion dataset and create a classifier to classify fashion clothing into categories.
- Objectives : Students should be able to classify movie reviews into positive and negative reviews.
- Prerequisite : 1. Basic of programming language.
  2. Concept of classification.
  3. Concept of DNN.
- Theory :

What is classification ?

- It is a type of supervised learning in ML that involves categorizing data into predefined classes or categories based on a set of features or characteristics.
- It is used to predict the class of new, unseen data based on the patterns learned from the labeled training data.

What is CNN ?

- CNN are commonly used for image classification tasks, and they are designed to automatically learn and extract features from input images.
- Let's consider an example of using a CNN to classify images of handwritten digits.
- The input to the network is an image of size 28 X 28 pixels, and the output is probability distribution over the 10 possible digits (0 to 9).

CNNs have a wide range of app. in :

1. Image classification
2. Object detection.
3. Semantic segmentation
4. NLP
5. Medical imaging
6. Auto. Vehicles.

How Deep Neural Network Work on classification using CNN :

- Input layer : The input layer of the network takes in the image data as input.
- Convolutional layer : It apply filters to the input images to extract relevant features.
- Activation functions : It is applied to the output of each convolutional layer to introduce non-linearity into the network.
- Pooling layers : It downsample the feature maps to reduce the spatial dimensions of the data.
- Dropout layer : It is used to prevent overfitting by randomly dropping out a percentage of neurons in the network during training.
- Fully connected layer : It takes the flattened o/p from the last pooling layer and perform the classification tasks.

- Conclusion : In this way we can classify fashion clothing into categories using CNN.