**PDC PROJECT PROPOSAL**

**Members: -**

* Dinesh Dhanjee (21K-3459)
* Ameer Hamza (21K-3463)
* Aneeq Muneer (21K-4582)

**Algorithms: -**

1. Brute Force Algorithm
2. Rabin karp Algorithm

**Brute Force Algorithm: -**

* Introduction: -

The Brute force algorithm helps to generate an outer boundary called convex hull of a set of points located randomly in a 2d plane. The convex hull encloses all the points in the set. It makes an angle with all the points other than itself and creates a line segment with the points it has the smallest angle with. It repeats this process for all the points until the last one connects to the first one. When that happens, the convex hull is created.

* OpenMP implementation: -

In OpenMP we can create as many threads as there are points. Each thread will be assigned a point from the set. That point will also have the number of that thread as it’s identification ID as well. Each thread will find the point it makes the least angle with and then will make a line segment. After the parallel work is done, we can combine all the results to form the complete convex hull.

* MPI implementation: -  
  In MPI we will make multiple subsets of the total set of points and assign a subset of points to each processor. Each process will apply the brute force approach and will find the local convex hull of the set of assigned set points and then in the end all the local convex hull can be combined to create the complete convex hull.

**Rabin Karp Algorithm: -**

* Introduction: -

Rabin Karp Algorithm searches for a pattern in a string through a hash function. It selects a substring of the same length as that of the pattern from the main string and calculates the hash function for that substring. If it matches the hash function value of the pattern, then the substring is checked if it is like the pattern or not. If not, then the process is continued until the string ends otherwise the indexes of the pattern are output to the screen.

* OpenMP implementation: -

In OpenMp we can create as many threads as there are substrings possible in the main string which have a length equal to that of the pattern. Then all the threads will run the Rabin Karp hash function and calculate the hash value for its substring and if it matches it will verify by comparing the string otherwise it will end without any output. The thread that will find the pattern in the string will output the indexes assigned to it and others won’t output anything.

* MPI implementation: -  
  In MPI we can divide the possible substrings between processes and whichever process finds a hash value match will first compare to verify it has found the correct pattern or not. If it has found the correct one, then it will output the indexes and the other processes will stop their task and won’t perform anymore tasks. If it has found the incorrect pattern with the same hash value, then the process will not output anything and carry on with the remaining tasks.