ROHIT RAJ

1RV17CS125

BATCH- C2

Program6 - K means

CODE:

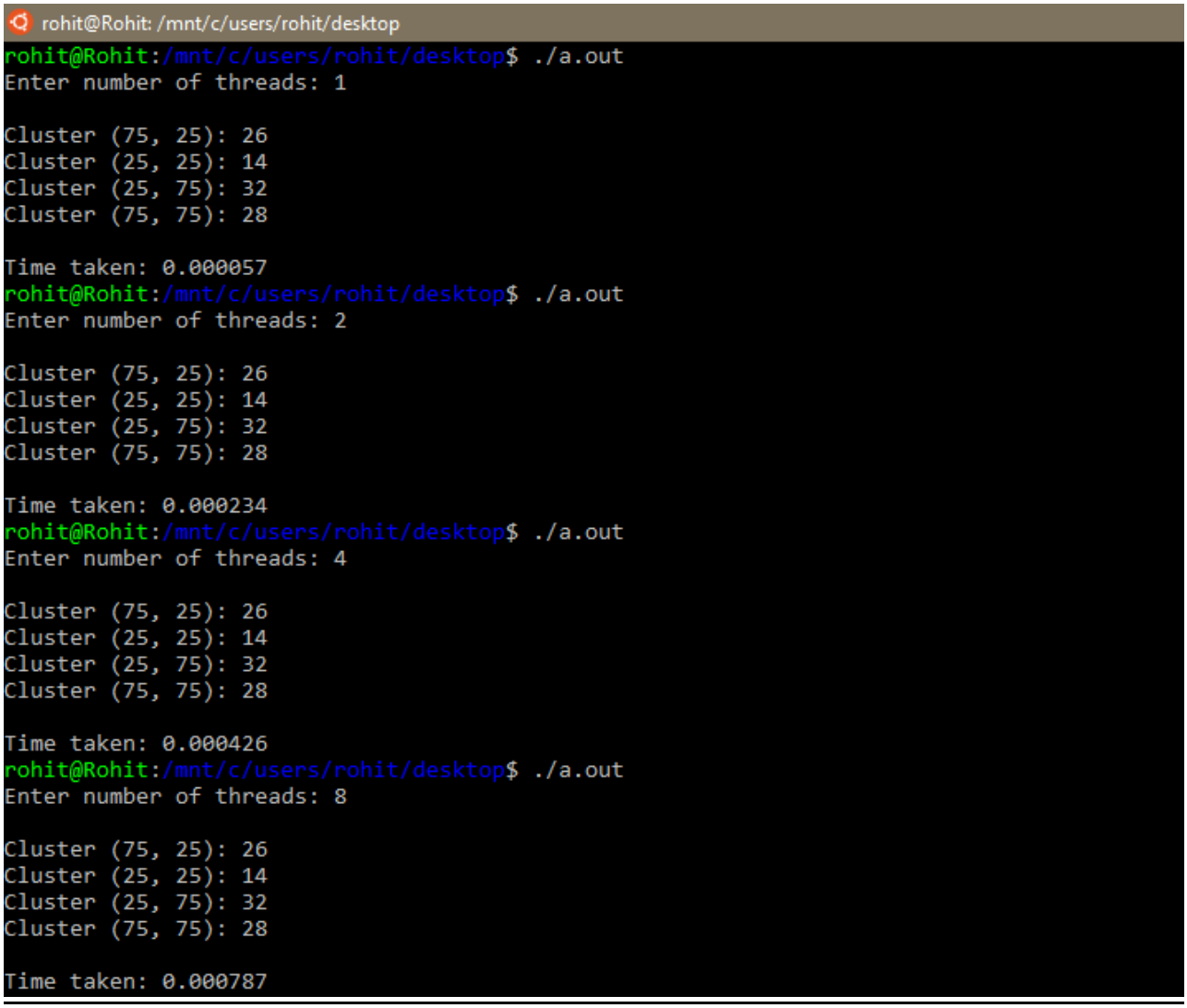
```
#include<stdio.h>
#include<omp.h>
#include<math.h>
#include<stdlib.h>
#define CLUSTER_SIZE 4
#define POINTS_SIZE 1000000
#define PRINT_POINTS 0

int cluster[CLUSTER_SIZE][2] = {{75, 25}, {25, 25}, {25, 75}, {75, 75}};
long long cluster_count[CLUSTER_SIZE];
int points[POINTS_SIZE][2];

void populate_points() {
        long long i;
        for(i = 0; i < CLUSTER_SIZE; i++) {
                cluster_count[i] = 0;
        }
        for(i = 0; i < POINTS_SIZE; i++) {
                srand(i);
```

```
            points[i][0] = rand() % 100;
            points[i][1] = rand() % 100;
        }
}

double get_distance(int x1,  int y1,  int x2,  int y2) {
        int x = x2-x1, y = y2-y1;
        return (double)sqrt((x * x) + (y * y));
}

int main() {
        double t;
        populate_points();
        long long i;
        if(PRINT_POINTS !=  0) {
            for(i = 0; i < CLUSTER_SIZE; i++) {
                printf("\nCluster %lld : (%d, %d)", i+1, cluster[i][0],
cluster[i][1]);
            }
            printf("\n\n");
        }

        int nt = 0;
        printf("Enter number of threads: ");
        scanf("%d", &nt);

        t = omp_get_wtime();
```

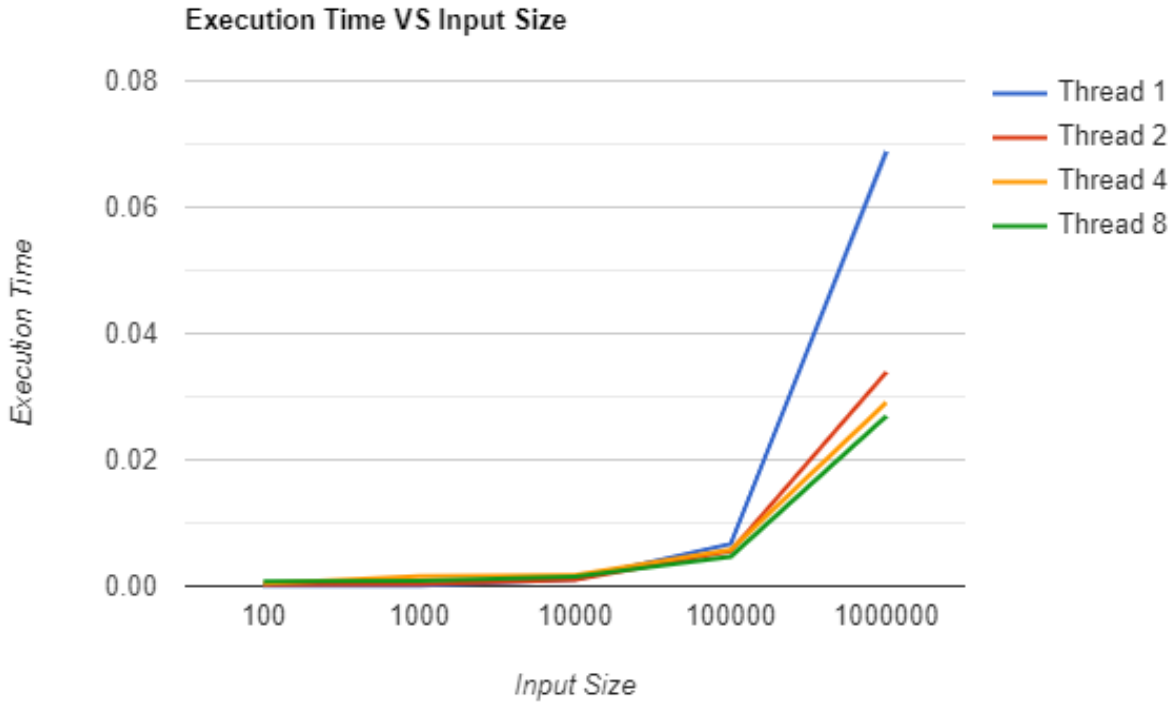
```
	#pragma omp parallel for private(i) shared(points, cluster) reduction(+:cluster_count) num_threads(nt)
	for(i = 0; i < POINTS_SIZE; i++) {
		double min_dist = 100, cur_dist = -1;
		int j, cluster_index = -1;
		for(j = 0; j < CLUSTER_SIZE; j++) {
			cur_dist = get_distance(points[i][0], points[i][1], cluster[j][0], cluster[j][1]);
			if(cur_dist<min_dist) {
				min_dist = cur_dist;
				cluster_index = j;
			}
		}
		if(PRINT_POINTS !=  0) {
			printf("\n(%d, %d) belongs to (%d, %d)", points[i][0], points[i][1], cluster[cluster_index][0], cluster[cluster_index][1]);
		}
		cluster_count[cluster_index]++;
	}
	t = omp_get_wtime() - t;
	for(i = 0; i < CLUSTER_SIZE; i++) {
		printf("\nCluster (%d, %d): %lld", cluster[i][0], cluster[i][1], cluster_count[i]);
	}
	printf("\n\nTime taken: %lf\n", t);
	return 0;
```

}

OUTPUT:


rohit@Rohit: /mnt/c/users/rohit/desktop
rohit@Rohit:/mnt/c/users/rohit/desktop$ ./a.out
Enter number of threads: 1
Cluster (75, 25): 26
Cluster (25, 25): 14
Cluster (25, 75): 32
Cluster (75, 75): 28
Time taken: 0.000057
rohit@Rohit:/mnt/c/users/rohit/desktop$ ./a.out
Enter number of threads: 2
Cluster (75, 25): 26
Cluster (25, 25): 14
Cluster (25, 75): 32
Cluster (75, 75): 28
Time taken: 0.000234
rohit@Rohit:/mnt/c/users/rohit/desktop$ ./a.out
Enter number of threads: 4
Cluster (75, 25): 26
Cluster (25, 25): 14
Cluster (25, 75): 32
Cluster (75, 75): 28
Time taken: 0.000426
rohit@Rohit:/mnt/c/users/rohit/desktop$ ./a.out
Enter number of threads: 8
Cluster (75, 25): 26
Cluster (25, 25): 14
Cluster (25, 75): 32
Cluster (75, 75): 28
Time taken: 0.000787

Graph:


Execution Time VS Input Size
0.08
0.06
0.04
0.02
0.00
Execution Time
100
1000
10000
100000
1000000
Input Size
Thread 1
Thread 2
Thread 4
Thread 8