```
%%writefile Guari_bubble.cu

#include <iostream>
#include <omp.h>
#include<chrono>//for calculating time
#include <bits/stdc++.h>
using namespace std::chrono;
using namespace std;
void sequentialBubbleSort(int *, int);
void parallelBubbleSort(int *, int);
void swap(int &, int &);
void sequentialBubbleSort(int *a, int n)
{
int swapped;
for (int i = 0; i < n; i++)
{
swapped = 0;
for (int j = 0; j < n - 1; j++)
{
if (a[j] > a[j + 1])
{
swap(a[j], a[j + 1]);
swapped = 1;
}
}
if (!swapped)
break;
}
}
void parallelBubbleSort(int *a, int n)
{
int swapped;
for (int i = 0; i < n; i++)
{
swapped = 0;
int first=i%2;
#pragma omp parallel for shared(a,first)
for (int j = first; j < n - 1; j++)
{
if (a[j] > a[j + 1])
{
swap(a[j], a[j + 1]);
swapped = 1;
}
}
if (!swapped)
break;
}
}
void swap(int &a, int &b)
{
int test;
test = a;
a = b;
b = test;
}
int main()
{
int *a, n;
cout << "\n enter total no of elements=>";
cin >> n;
a = new int[n];
cout << "\n enter elements=>";
for (int i = 0; i < n; i++)
{
cin >> a[i];
}
// Sequential algorithm
auto start = high_resolution_clock::now();
sequentialBubbleSort(a, n);
auto stop = high_resolution_clock::now();
auto seq_time = duration_cast<microseconds>(stop - start);
cout << "\n sorted array is=>";
for (int i = 0; i < n; i++)
{
cout << a[i] << endl;
}
cout << "\nSequential Time: " << seq_time.count() << endl;
auto start_time = high_resolution_clock::now();
parallelBubbleSort(a, n);
auto end_time = high_resolution_clock::now();
auto par_time = duration_cast<microseconds>(end_time - start_time);
cout << "\n sorted array is=>";
for (int i = 0; i < n; i++)
{
cout << a[i] << endl;
}
cout << "\nParallel Time: " << par_time.count()<< endl;
delete[] a;
return 0;
}
```

```
Writing Guari_bubble.cu
```

```
!nvcc Guari_bubble.cu -o Guari_bubble
```

```
!./Guari_bubble
```

```
 enter total no of elements=>5

 enter elements=>41
74
1
86
23

 sorted array is=>1
23
41
74
86

Sequential Time: 0

 sorted array is=>1
23
41
74
86

Parallel Time: 0
```

Start coding or generate with AI.