Laboratory Practice 1 :
High Performance Computing Mini Project
Parallel Bubble Sort

Project ID: 13

Project created by :
Prathamesh Chaudhari (50)
Sanket Sutar(49)
Rupesh Deshmukh(52)
Piyush Patil(51)

Project Guide: Prof. J. R. Mankar

December 24, 2020

Contents

List of Figures

Chapter 1

Problem Statement

To implement Parallel bubble Sort using OpenMP API.

.

Chapter 2

Objectives

- To implement parallel bubble sort by using OpenMp.
- To implement sequential bubble sort.
- To compare parallel bubble sort with sequential bubble sort.

Chapter 3

Bubble Sort

Bubble sort is a simple sorting algorithm. This sorting algorithm is comparison-based algorithm in which each pair of adjacent elements is compared and the elements are swapped if they are not in order. This algorithm is not suitable for large data sets as its average and worst case complexity are of (n2) where n is the number of items.

```
procedure bubbleSort( list : array of items )

   loop = list.count;

   for i = 0 to loop-1 do:
      swapped = false

      for j = 0 to loop-1 do:

         /* compare the adjacent elements */
         if list[j] > list[j+1] then
            /* swap them */
            swap( list[j], list[j+1] )
            swapped = true
         end if

      end for

      /*if no number was swapped that means
      array is sorted now, break the loop.*/

      if(not swapped) then
         break
      end if

   end for

end procedure return list
```

Figure 3.1: Pseudocode of BubbleSort

Chapter 4

Parallel Bubble sort

Implemented as a pipeline.

```
void *Parallel_bubble_sort(void *arg)
{
   int id, i;
   Get_id(id);
   int lsize = size/no_threads;
   if (size % no_threads != 0)
      lsize++;
   int my_start = id*lsize;
   int my_end = min(size-1, (id+1)*lsize);
   for (i=-id; i<size+no_threads-id; i++)
      if (i >= 0 && i<size)
         Local_loop(my_start, my_end);
      Barrier(no_threads);
   }
   if (id == no_threads-1)
      Output_array();
}
```

Figure 4.1: Pseudocode of Parallel BubbleSort

Chapter 5

OpenMP

OpenMP is a widely adopted shared memory parallel programming interface providing high level programming constructs that enable the user to easily expose an application's task and loop level parallelism in an incremental fashion. The range of OpenMP applicability was significantly extended recently by the addition of explicit tasking features. The OpenMP is the dominant programming model for heterogeneous systems and adopted by Intel, Clear Speed, PGI and CAPS SA. The idea behind OpenMP is that the user specifies the parallelization strategy for a program at a high level by providing the program code.

Chapter 6

Screenshot of Output

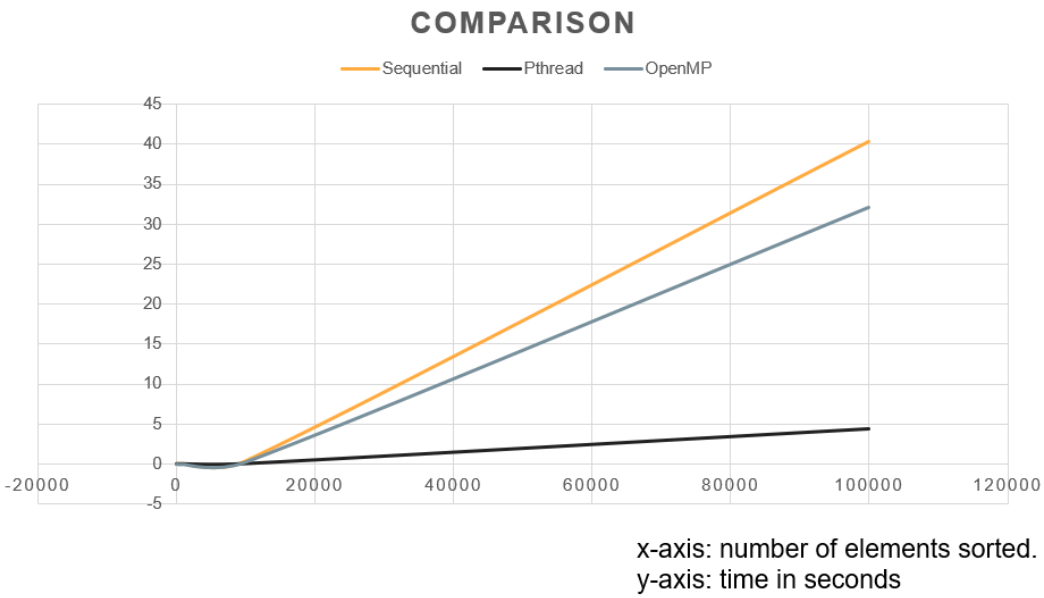

Figure 6.1: Number of elements sorted vs Time

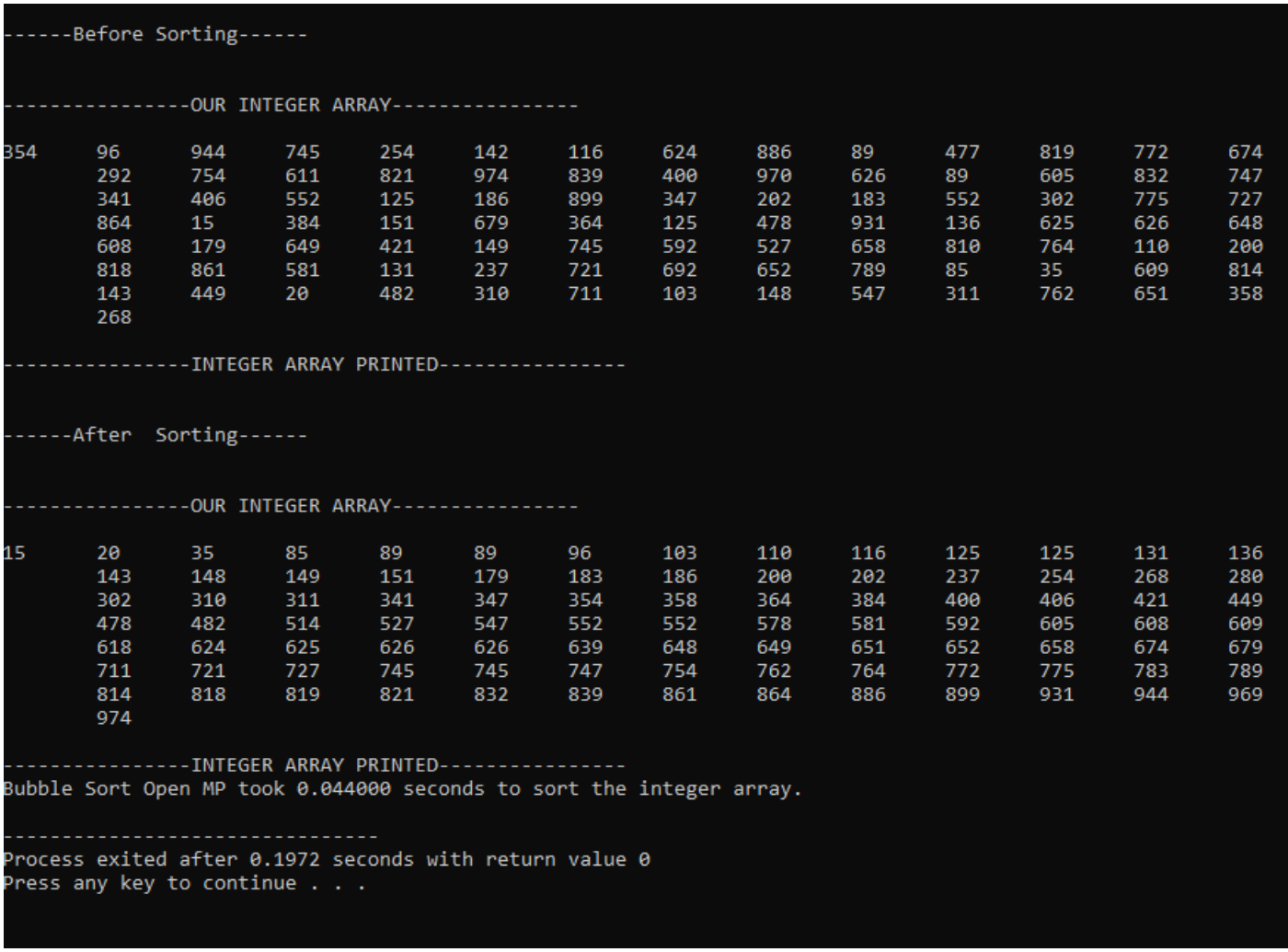

Figure 6.2: Output

Chapter 7

Outcomes

- The sequential bubble sort is inefficient sorting method for common usage.
- For large arrays parallel algorithms perform far better than sequential algo-rithms.
- parallel algorithms has better CPU utilization for large arrays.

Chapter 8

Conclusion

Hence we have successfully implemented parallel bubble Sort by applying parallelism using OpenMP constructs.