2-parallel-bubble

May 8, 2025

[11]:
```
%%writefile bubble_sort.cpp
#include<iostream>
#include<omp.h>

using namespace std;

void bubble(int array[], int n){
    for (int i = 0; i < n - 1; i++){
        for (int j = 0; j < n - i - 1; j++){
            if (array[j] > array[j + 1])
                swap(array[j], array[j + 1]);
        }
    }
}

void pBubble(int array[], int n){
    for(int i = 0; i < n; ++i){
        // Odd phase
        #pragma omp for
        for (int j = 1; j < n; j += 2){
            if (array[j] < array[j - 1])
                swap(array[j], array[j - 1]);
        }

        #pragma omp barrier

        // Even phase
        #pragma omp for
        for (int j = 2; j < n; j += 2){
            if (array[j] < array[j - 1])
                swap(array[j], array[j - 1]);
        }

        #pragma omp barrier
    }
}
```

```
void printArray(int arr[], int n){
    for(int i = 0; i < n; i++) cout << arr[i] << " ";
    cout << "\n";
}

int main(){
    int n = 10;
    int arr[n];
    double start_time, end_time;

    for(int i = 0, j = n; i < n; i++, j--) arr[i] = j;

    // Sequential Bubble Sort
    start_time = omp_get_wtime();
    bubble(arr, n);
    end_time = omp_get_wtime();
    cout << "Sequential Bubble Sort took : " << end_time - start_time << "␣
 ↪seconds.\n";
    printArray(arr, n);

    for(int i = 0, j = n; i < n; i++, j--) arr[i] = j;

    // Parallel Bubble Sort
    start_time = omp_get_wtime();
    #pragma omp parallel
    {
        pBubble(arr, n);
    }
    end_time = omp_get_wtime();
    cout << "Parallel Bubble Sort took : " << end_time - start_time << "␣
 ↪seconds.\n";
    printArray(arr, n);

    return 0;
}
```

```
Writing bubble_sort.cpp
```

[12]:
```
!g++ -fopenmp bubble_sort.cpp -o bubble_sort
```

[13]:
```
!./bubble_sort
```

```
Sequential Bubble Sort took : 4.304e-06 seconds.
1 2 3 4 5 6 7 8 9 10
Parallel Bubble Sort took : 0.000154284 seconds.
1 2 3 4 5 6 7 8 9 10
```