2-parallel-merge

May 8, 2025

[16]:
```
%%writefile merge_sort.cpp
#include <iostream>
#include <omp.h>

using namespace std;

void merge(int arr[], int low, int mid, int high) {
    int n1 = mid - low + 1;
    int n2 = high - mid;

    int left[n1], right[n2];
    for (int i = 0; i < n1; i++) left[i] = arr[low + i];
    for (int j = 0; j < n2; j++) right[j] = arr[mid + 1 + j];

    int i = 0, j = 0, k = low;
    while (i < n1 && j < n2) {
        if (left[i] <= right[j])
            arr[k++] = left[i++];
        else
            arr[k++] = right[j++];
    }

    while (i < n1) arr[k++] = left[i++];
    while (j < n2) arr[k++] = right[j++];
}

void parallelMergeSort(int arr[], int low, int high) {
    if (low < high) {
        int mid = (low + high) / 2;

        #pragma omp parallel sections
        {
            #pragma omp section
            parallelMergeSort(arr, low, mid);

            #pragma omp section
            parallelMergeSort(arr, mid + 1, high);
```

```
        }

        merge(arr, low, mid, high);
    }
}

void mergeSort(int arr[], int low, int high) {
    if (low < high) {
        int mid = (low + high) / 2;
        mergeSort(arr, low, mid);
        mergeSort(arr, mid + 1, high);
        merge(arr, low, mid, high);
    }
}

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;

    start_time = omp_get_wtime();
    mergeSort(arr, 0, n - 1);
    end_time = omp_get_wtime();
    cout << "Time taken by sequential algorithm: " << end_time - start_time <<␣
 ↪" seconds\n";

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

    start_time = omp_get_wtime();
    parallelMergeSort(arr, 0, n - 1);
    end_time = omp_get_wtime();
    cout << "Time taken by parallel algorithm: " << end_time - start_time << "␣
 ↪seconds\n";

    return 0;
}
```

```
Overwriting merge_sort.cpp
```

[17]:
```
!g++ -fopenmp merge_sort.cpp -o merge_sort
```

[18]:
```
!./merge_sort
```

```
Time taken by sequential algorithm: 2.642e-06 seconds
Time taken by parallel algorithm: 0.00203123 seconds
```