Threaded Merge Sort

HPP: Inleveropdracht 3

**Amaryllis Lee**

2 March 2020

Studentnumber : 1735290

# 

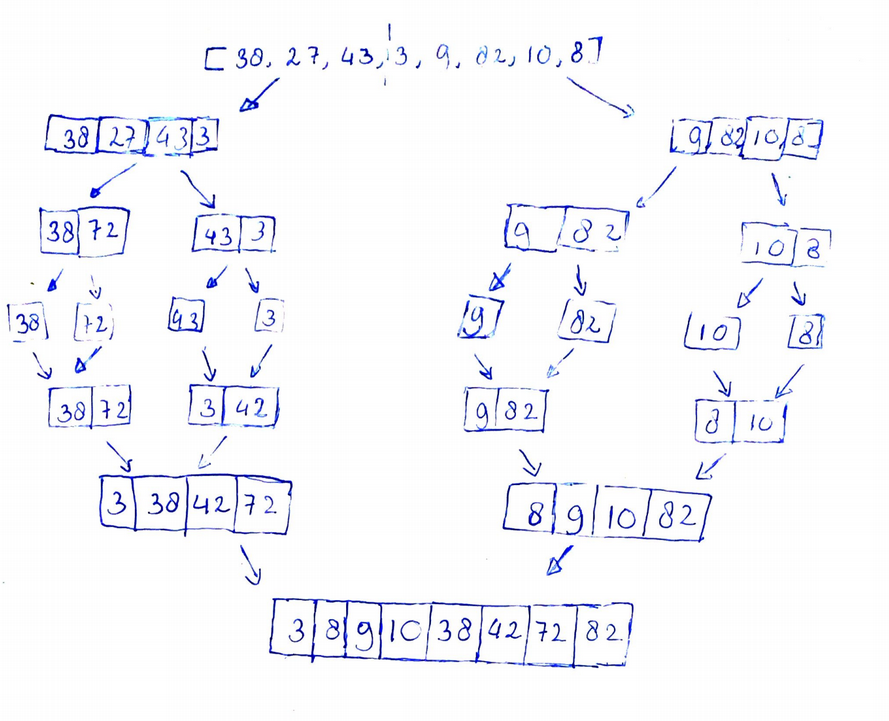
In this assignment, we are going to be parallelising the merge sort algorithm.  
*Merge sor*t is a ‘divide and conquer sorting algorithm. In merge sort, there are a series of three steps: *divide, conquer and combine*. Initially, divide the given array consisting of n elements into two parts of n/2 elements each. Sort the left part and right part of the array recursively. Merge the sorted left part and the right part to get a single sorted array  
Source: <https://www.researchgate.net/figure/Multithread-Merge-Sort_fig2_322065892>

# Design a single thread and multithreaded merge sort

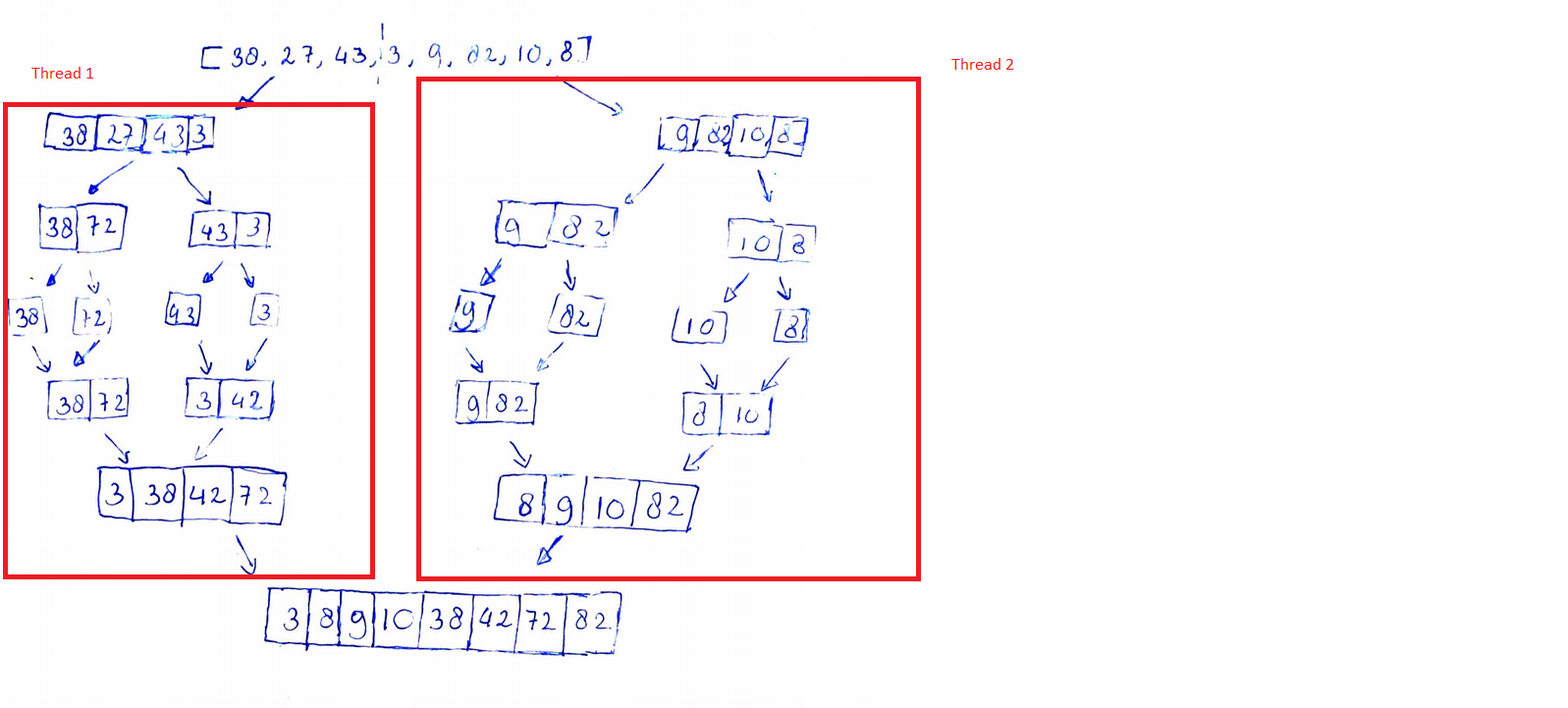
Design on paper (based on an example of an array of eight numbers) how you can sort a list using 1, 2, 4, 8 cores/processes/threads via the merge sort algorithm.

*Think carefully about distributing* ***the data across the different processes*** *and efficiently bringing the results together again. Make use of thread pools.*

## Single Thread

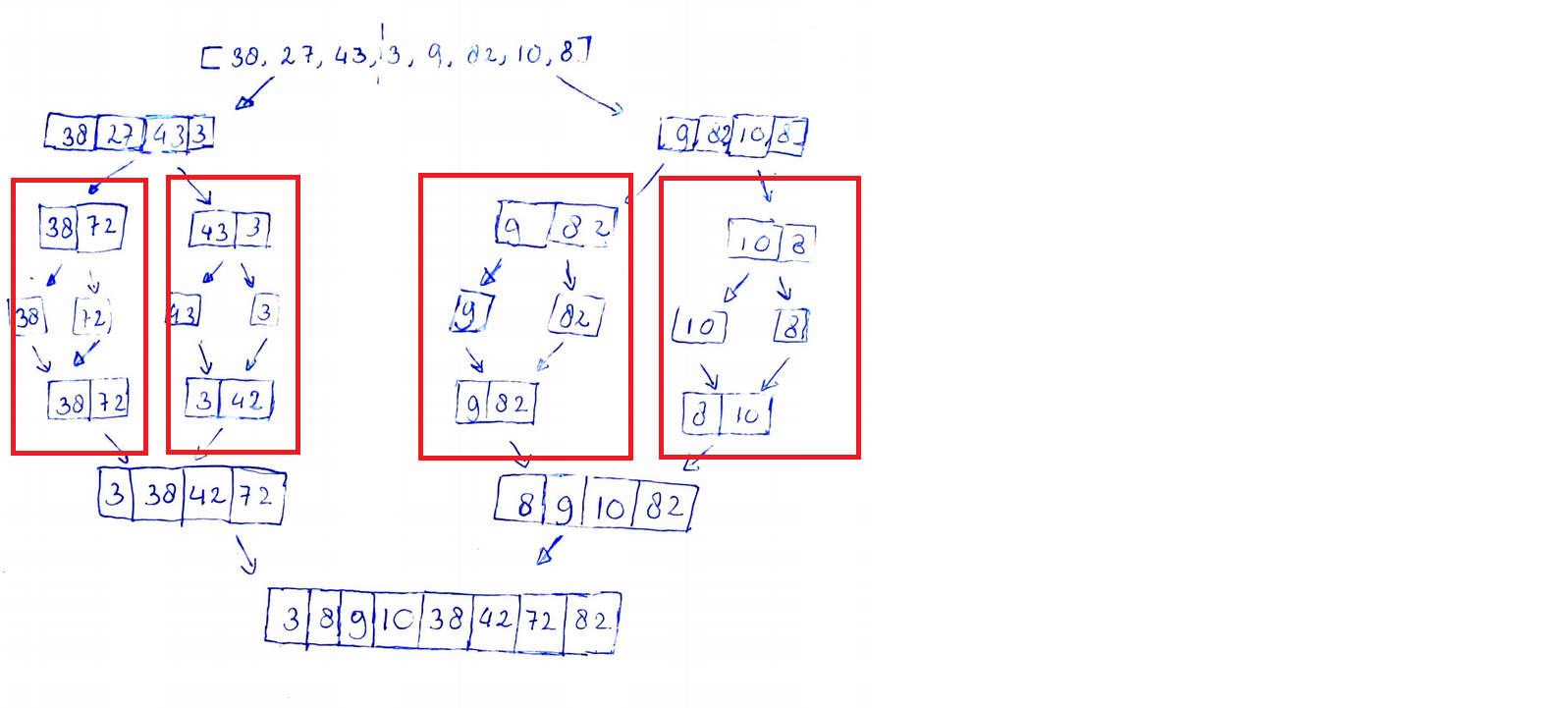
A single thread ( 1 thread ) is the amount of work completed by some software that runs as a single stream of instructions in a certain amount of time.  
The size for a single thread would be eight because we will be executing the whole array.  
Example:  


## Multithread: 2 thread/ core

The example on the left is what I assumed would be how you would sort this array with two threads. The data distribution (The size of the arrays per thread )would be 8/2 = 4.

Each thread is performing the same task/merge sort. The arrays would be split into subarrays, run merge sort on these subarrays. Eventually, the mergesort will merge the arrays.  
This is equivalent to *data parallelism.*

## Multithreading: 4 threads/cores

Data parallelism is also taking place for the four-thread merge sort. The merge sort divides the split lists again into subarrays of size 1 and merges the subarrays into sorted arrays of size 2.

The data distribution (The size of the arrays per thread ) would be 8/4 = 2.

# Implementation

The implementation of the multithreading mergesort is implemented in Python 3 using the *threading module.* The implementation can be found in my Github repository: [*https://github.com/AmaryllisLee/HPP*](https://github.com/AmaryllisLee/HPP)