# High Performance Computing for Data Science Project

Prof. Fiore Sandro Luigi

Guidolin Davide
davide.guidolin@studenti.unitn.it

Zanolli Giacomo
giacomo.zanolli@studenti.unitn.it

***Abstract*—This is the report for the High Performance Computing for Data Science course project.**

**The goal of the project was to implement a parallel solution for the *closest pair of points* problem and evaluate it on the HPC@UniTrento cluster.**

**In the following sections we will present the problem and we will describe some serial solutions for it. Then we will present how we implemented a parallel solution and finally we will present and discuss the evaluation of the parallel solution on the HPC cluster.**

## I. Introduction

The *parallel closest pair of points* problem consists in finding the smaller distance between two points in the plane. We will focus on the 2-dimensional problem, however, solutions for the $d$ dimensional space exist.

The naive solution to this problem would be to take each point in the plane and calculate the distance between that point and all the other points. The time complexity for this solution is $O(N^2)$ where $N$ is the number of points in the space. Better solutions have been proposed, in particular in 1976 Rabin proposed a randomized algorithm with an expected run time of $O(N)$ and in 1979 Fortune and Hopcroft proposed a deterministic $O(NloglogN)$ solution [1] assuming that the floor operation takes constant time.

In this project we will explore a divide and conquer approach with $O(N(logN)^2)$ time complexity. We chose it because divide and conquer is highly parallelizable so it will be easier to exploit the computational power of the HPC cluster.

## II. Problem analysis

Serial implementation

### A. Divide

The first step of the divide and conquer algorithm is the division of the original problem in sub problems.

The original set of points will be split in two subsets and each subset will be split in two subsets. This process will be repeated until we have three or less points in each subset.

### B. Find the closest pair

To find the closest pair in each subset we only have to perform three comparison and select the pair of points with the smallest distance in the subset.

### C. Merge

### D. Implementation

## III. Main steps towards parallelization

- Design of the parallel solution
- Implementation
- Benchmark on the HPC@UniTrento cluster

## IV. Final discussion

Conclusions

## References

[1] S. Fortune and J. Hopcroft, “A note on rabin’s nearest-neighbor algorithm,” *Information Processing Letters*, vol. 8, no. 1, pp. 20–23, 1979.