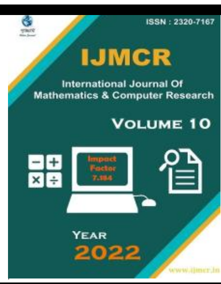
**International Journal of Mathematics and Computer Research**

**ISSN: 2320-7167**

**Volume 10 Issue 01 January 2022, Page no. – 2540-2541**

**Index Copernicus ICV: 57.55, Impact Factor: 7.184**

**DOI: 10.47191/ijmcr/v10i1.03**




# Closest Pair Algorithms in 2D Space, a Commentary on Complexity and Reductions

**Anand Sunder**
Master of Science, Texas Tech

| ARTICLE INFO | ABSTRACT |
|---|---|
| Published Online: 05 January 2022<br><br>Corresponding Author: **Anand Sunder** | One of the most challenging problems in computational geometry is closest pair of points given n points. Brute force algorithms[1] and Divide and conquer[1] have been verified and the lowest complexity of $O(nlog(n))$ attributed to latter class of algorithms, with worst case being for the former being $O(n^2)$. We propose a method of partitioning the set of n-points based on the least area rectangle that can circumscribe these points. |

**KEYWORDS:** Metric Spaces, Existence, Uniqueness, Differential Equations, MATLAB.

## INTRODUCTION

We circumscribe a rectangle for the set of points $S_n = \{(x_1,y_1),\ldots.(x_n,y_n)\}$ with smallest possible area, partition it into k parts width and length wise such that $k^2$ number of cells result.

**Hypothesis 1:** Candidate for closest pair identification is the cell with maximum density $\rho_{kmax} = Max\ (N_{kl}/A_k)$, where, $A_k = |(x_{max} - x_{min})(y_{max} - y_{min})|/k^2$, $l$ is the $l^{th}$ cell and minimum is found here.

**Hypothesis 2:** Candidate for closest pair identification is the cell with minimum density $\rho_{kmin} = Min\ (N_{kl}/A_k)$, where, $A_k = |(x_{max} - x_{min})(y_{max} - y_{min})|/k^2$, $l$ is the $l^{th}$ cell, minimum is found here.

**Hypothesis 3:** Candidate for closest pair identification is the cell with a density $\rho_k = N_{kl}/A_k$,where, $A_k = |(x_{max} - x_{min})(y_{max} - y_{min})|/k^2$, $l$ is the $l^{th}$ cell, minimum is found here.

**Problem**:

Find $\{(x_i,y_i),(x_j,y_j)\}$ S.T $d_{ij}(\,distance)$ is a minimum.

$\sqrt{d_{ij}} = (x_i - x_j)^2 + (y_i - y_j)^2$

$\frac{log d_{ij}}{2} = log(x_i - x_j) + \log(y_i - y_j)$

Formulating an objective function or Reduction that solves the easier problem

$$\text{Minimize } Z = |log(x_i - x_j)| + |log(y_i - y_j)|$$

$$\forall \quad x_{min} < x_i < x_{max}$$
$$x_{min} < x_j < x_{max}$$
$$y_{min} < y_i < y_{max}$$
$$y_{min} < y_j < y_{max}$$

Let X=$|log(x_i - x_j)|$ and Y=$|log(y_i - y_j)|$

Z=X+Y

| X-Y| >0

0<X<$|log(x_{max} - x_{min})|$

0<Y<$|log(y_{max} - y_{min})|$

Following the conventional linear programming this would turn out to be an NP hard problem, i.e $O(n^2)$ or higher orders.

With the improved solver proposed by Cohen e.t al [2], it takes O∗($n^\omega$log(n/δ)) time where ω is the exponent of matrix multiplication and δ is the relative accuracy.

Dividing the smallest circumscribing rectangle into $k^2$ cells, assuming minimum is found for the $l^{th}$ cell.

Although when minimum computed for individual cell by divide and conquer algorithm[1] results in a complexity of $O(nlog(n))$ , with $k^2$ iterations Recurrence relation of complexity becomes $T(n) = k^2 O(n^\omega$log(n/δ)) specifically if Hypothesis 3 holds true

In real scenarios either of three hypotheses hold true, but for cases with Hypothesis 2 or 1, We could get the complexity to $O * (n^\omega$log(n/δ)), here the input size n <<$n_T$ where $n_T$ is the sample size.

For the formulation above $\omega = 1$ and

By iteratively increasing partition size $k = 1,2,3,\ldots$ we can reduce the absolute overall complexity.

Although it's impossible to theoretically reduce the complexity beyond $O(n^\omega$log(n/δ)) we can enhance the divide and conquer strategies based on density distribution of partitioned points in a 2-D space for reduced overall compute time[3].

## REFERENCES


1. Subash Suri, Closest Pair Problem: https://sites.cs.ucsb.edu/~suri/cs235/ClosestPair.pdf
2. Michael B. Cohen, Yin Tat Lee, Zhao Song, 2020, Solving Linear Programs in the Current Matrix Multiplication Time: https://arxiv.org/abs/1810.07896
3. Meysam Aghighi ,2017, Computational Complexity of some Optimization Problems in Planning: https://www.divaportal.org/smash/get/diva2:1087096/FULLTEXT03
4. Brute Force Closest Pair and Convex-Hull: http://www.csl.mtu.edu/cs4321/www/Lectures/Lecture%206%20%20Brute%20Force%20Closest%20Pair%20and%20Convex%20and%20Exhausive%20Search.html