Finding the closest pair of points

COMP 5703 - KATIE DUONG

The Problem

- Given the set of n points on the plane S.
- Compute a closet pair (P,Q) in S such that:

$$d(P,Q) = \min\{d(p,q) : p,q \in S,\ p \neq q\}$$

Solution

Present randomized incremental algorithm store the points in a grid.

The worst-case running time is $O(n^2 \log n)$.

Using hashing table with expected time improves to O(n).

Outline

- Algorithm
- What is d-grid and how to store points in it?
- How to compute closet pair distance of set S?
- How many points in one cell?
- How the total running time is improve?
- The randomized algorithm

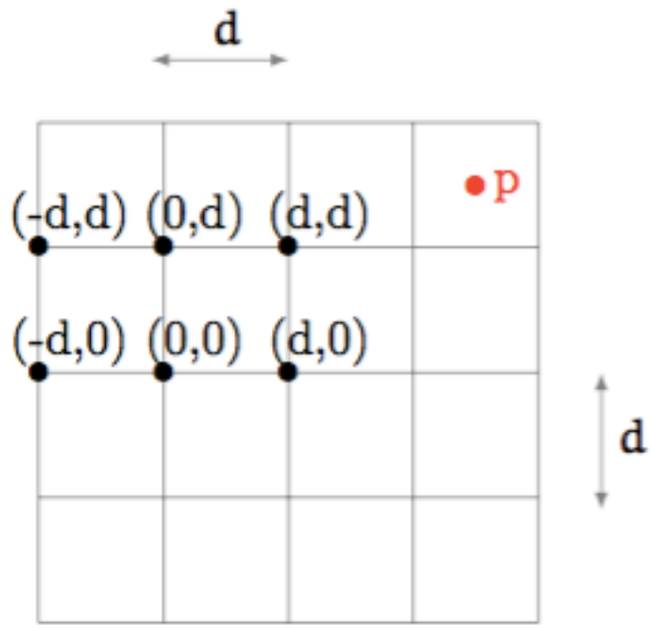
Algorithm

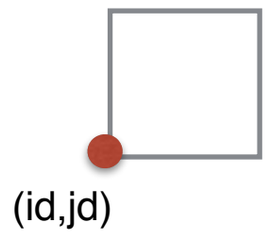
Let $S = \{p_1, p_2, \ldots, p_n\}$ be a set of n points in the plane. The algorithm computes a closest pair in the set $S_i = \{p_1, p_2, \ldots, p_i\}$ for $i = 2, 3, \ldots, n$.

1. $i := 2$; CP-distance $= d(p_1, p_2)$;
2. for $i = 3, 4, \ldots, n$, given the CP-distance d of S_{i-1}, compute the CP-distance of S_i.

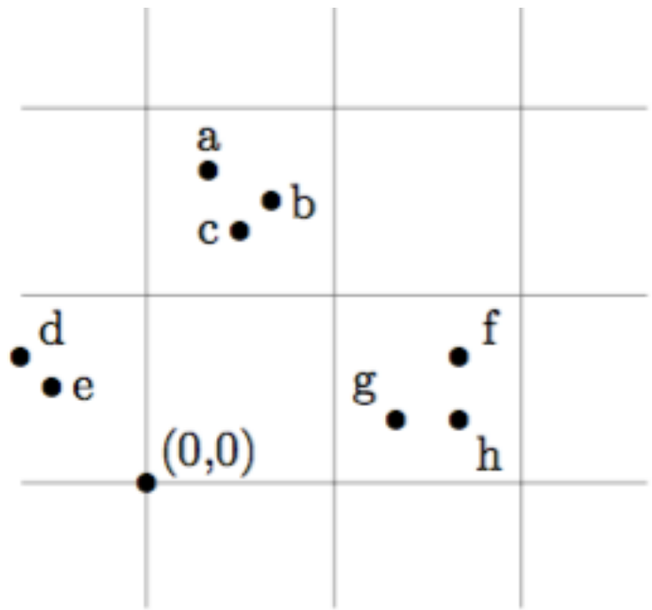
In the for-loop, we store the points of S_{i-1} in a d-grid. To compute the CP-distance of S_i, we compute the distance beween p_i and all points in the 9 cells around p_i. If the CP-distance does not change, we add p_i to the d-grid. If the CP-distance changes, we store all points of S_i in a new grid.

What is d-grid how to store points in it?



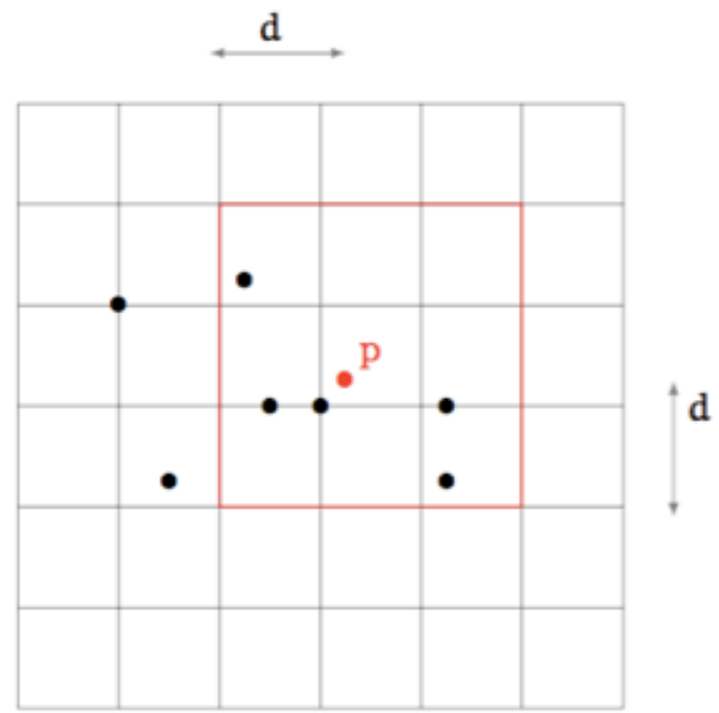


The $\mathtt{id}$ of a cell with lower-left corner $(\mathtt{id}, \mathtt{jd})$ is $(\mathtt{i}, \mathtt{j})$. The point $\mathtt{p} = (\mathtt{p}_x, \mathtt{p}_y)$ is in the cell with $\mathtt{id} = (\lfloor \mathtt{p}_x/\mathtt{d} \rfloor, \lfloor \mathtt{p}_y/\mathtt{d} \rfloor)$.



How to compute closet pair distance of set S?

Store points of S_{i-1} in d-grid



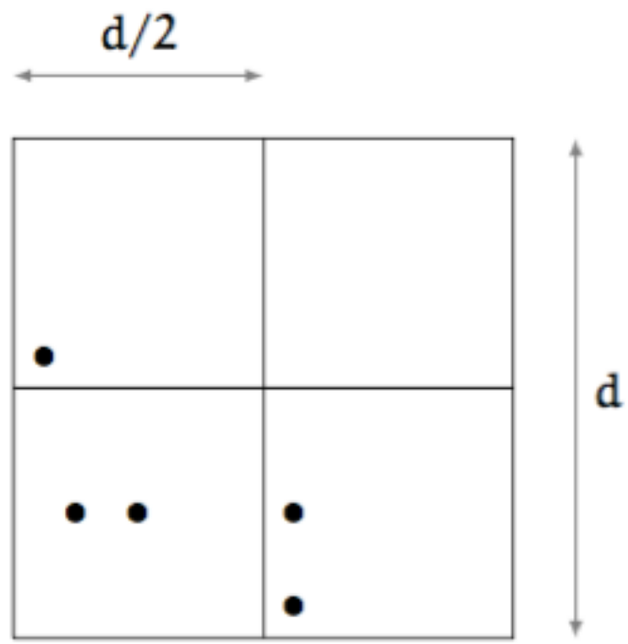
Let $S = \{p_1, p_2, \ldots, p_n\}$ be a set of n points in the plane. The algorithm computes a closest pair in the set $S_i = \{p_1, p_2, \ldots, p_i\}$ for $i = 2, 3, \ldots, n$.

1. $i := 2$; CP-distance $= d(p_1, p_2)$;

2. for $i = 3, 4, \ldots, n$, given the CP-distance d of S_{i-1}, compute the CP-distance of S_i.

In the for-loop, we store the points of S_{i-1} in a d-grid. To compute the CP-distance of S_i, we compute the distance beween p_i and all points in the 9 cells around p_i. If the CP-distance does not change, we add p_i to the d-grid. If the CP-distance changes, we store all points of S_i in a new grid.

How many points in one cell?



How the total running time is improve?

Total running time after n interation:

binary search tree : $O(n^2 \log n)$

hasing table : $O(n^2)$

If we need to make new grid, for example

iteration 1: store p_1, then add p_2 we get CP-distance = (p_1, p_2).

iteration 2: add p_3 we get CP-distance = (p_2, p_3).

iteration 3: add p_4 we get CP-distance = (p_3, p_4).

iteration 4: add p_5 we get CP-distance = (p_4, p_5).

If we do not need to make new grid by changing the order of point to store in the grid as below.

iteration 1: store p_1, then add p_2 we get CP-distance = (p_1, p_2).

iteration 2: add p_3 we get CP-distance = (p_1, p_2).

iteration 3: add p_4 we get CP-distance = (p_1, p_2).

iteration 4: add p_5 we get CP-distance = (p_1, p_2).

The randomized algorithm

$$X_i = \begin{cases} 1 & \text{if the grid changes in iteration } i \\ 0 & \text{if the grid does not change in iteration } i \end{cases}$$

$$X = O(n) + \sum_{i=3}^{n} X_i \cdot O(i)$$

$$E(X) = O(n) + \sum_{i=3}^{n} E(X_i) \cdot O(i).$$

$$E(X_i) = \Pr(X_i = 1).$$

$$E(X_i) = \Pr(X_i = 1) = 2/i.$$

$$E(X) = O(n) + \sum_{i=3}^{n} \frac{2}{i} \cdot O(i) = O(n) + \sum_{i=3}^{n} O(1) = O(n).$$

use a binary search tree

$$X = O(n \log n) + \sum_{i=3}^{n} X_i \cdot O(i \log i)$$

$$E(X) = O(n \log n).$$

References

[1] Sariel Har-Peled. *Geometric Approximation Algorithms.* American Mathematical Society.

[2] Jon Kleinberg, Eva Tardos, *Algorithm Design*, Chapter 13. Addison Wesley.

[3] Michiel Smid. *The closest pair problem: a randomized incremental algorithm.* Carleton University

[4] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. *Introduction to Algorithms.* The MIT Press Cambridge, Massachusetts London, England.

[5] Mordecai Golin, Rajeev Raman, Christian Schwarz, Michiel Smid. *Simple randomized algorithms for closest pair problems.* Nordic Journal of Computing, 2, 1995, pages 3-27.