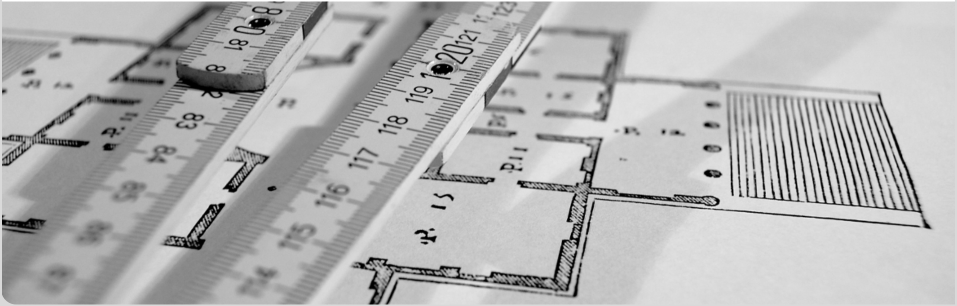
Karlsruhe Institute of Technology

Parallel Algorithm for Closest Pair Problem

Ge Wu | July 23, 2013

INSTITUTE FOR THEORETICAL INFORMATICS



Problem Description

Closest Pair Problem

- Given n **different** unordered points $P = \{p_1, p_2, ..., p_n\}$ in **unit square**

$$p_i = (x_i, y_i) \in (0,1) \times (0,1) \subset \mathbb{R}^2$$

- Find a pair of points with closest euclidean distance between them
- Find any pair if there's a tie.

Background

- $O(n \log n)$ lower bound in comparison tree model
- **Bentley and Shamos 1976**
  $O(n \log n)$ algorithm using divide and conquer
- **Rabin 1976**
  $O(n)$ randomized algorithm with $O(1)$ floor function
- **Fortune and Hopcroft 1978**
  Deterministic $O(n \log \log n)$ algorithm with $O(1)$ floor function
- **Khuller and Matias 1995**
  Another O(n) randomized algorithm

Sample

- Compute all pair of distances: $O((n^{\frac{2}{3}})^2) = O(n^{\frac{4}{3}})$
- Divide & Conquer: $O(n^{\frac{2}{3}} \log n^{\frac{2}{3}}) = O(n^{\frac{2}{3}} \log n) \subset O(n)$
  More samples possible: $O(\frac{n}{\log n} \log \frac{n}{\log n}) \subset O(n)$
- Other approach?

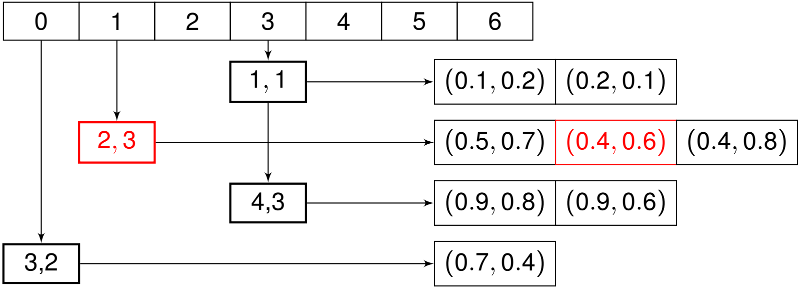
Partition

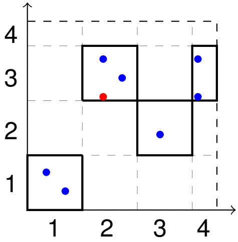
- Map points to the cells
- Divide the coordinates by cell length and truncate them to integer
- Partition $\mathbb{D} = \{D_1, D_2, ..., D_k\}$ with:

$$\bigcup_{i=1}^{k} D_i = P, D_i \cap D_j = \emptyset, k <= n$$

- $O(n)$ with hashing

Partition





$$\boldsymbol{H}(cell_{x\_idx}, cell_{y\_idx}) = (cell_{x\_idx} + 2 \cdot cell_{y\_idx}) \mod 13$$

Compute

Compute all pairs distances

$$O(\sum_{i=1}^{k} f_i \cdot (f_i + |neighbor(G_i)|)) = O(\sum_{i=1}^{k} f_i^2) = O(n)$$

Theorem

Let $\mathbf{f_i} = |G_i|$ *with* $1 \leq i \leq k$ *be the number of points in* i*-th non-empty cell and* c, d *some positive constants, then*

$$Prob\left(\sum_{i=1}^{k} f_i^2 <= c \cdot n\right) \geq 1 - \frac{1}{2^{n^d}}$$

Alternative for Sampling

Karlsruhe Institute of Technology

Recursively calculate the closest distance on samples $S = \{s_1, ..., s_{n^{\frac{2}{3}}}\}$

- Sample another $n^{\frac{4}{9}}$ points S' from S
- Get closest distance on S' by calculating all pair distances: $O(n^{\frac{8}{9}})$
- Partition and compute the closest distance on S

Pseudocode

0 Cell set $H = \emptyset$, Result $rst = \infty$

1 Choose set S of $n^{\frac{2}{3}}$ samples $O(n^{\frac{2}{3}})$

$\delta = min_{x,y \in S, x \neq y}(dist(x, y))$ $O(n^{\frac{8}{9}})$

2 **for** $i = 1$ **to** n **do** $O(n)$

  $c = cell(p_i)$

  if $not\ H.findCell(c)$ **then**

    $H.addCell(\{c, idx = \{c.x, c.y\}\})$

  $H.findCell(c).addPoint(p_i)$

3 **foreach** c_1 **in** H **do** $O(n)$

  foreach c_2 **in** $\{c_1\} \cup neighbor(c_1)$ **do**

    foreach (p_i, p_j) **in** $c_1 \times c_2$

      $rst = min(rst, dist(p_i, p_j))$

Runtime: $O(n)$ with high probability

Worst case: $O(n^2)$

Parallelization

0 Cell set $H = \emptyset$, Result $rst = \infty$ ## H synchronized Hashmap

1 Choose set S of $n^{\frac{2}{3}}$ samples ## allocate p PEs — $O(n^{\frac{2}{3}}/p)$

$\delta = min_{x,y \in S, x \neq y}(dist(x,y))$ ## recurse once — $O(n^{\frac{8}{9}} + \log p)$

2 **for** $i = 1$ **to** n **do** ## allocate p PEs — $O(n/p)$

$c = cell(p_i)$

if *not* $H.findCell(c)$ **then**

$H.addCell(\{c, idx = \{c.x, c.y\}\}$ ## maximal n times

$H.findCell(c).addPoint(p_i)$

3 **foreach** c_1 **in** H **do** — $O(n/p + \log p)$

foreach c_2 **in** $\{c_1\} \cup neighbor(c_1)$ **do**

foreach (p_i, p_j) **in** $c_1 \times c_2$ ## allocate $\lfloor p \cdot \frac{\#Pairs}{|c_1||c_2|} \rfloor$ PEs

$rst = min(rst, dist(p_i, p_j))$

Total runtime: $O(\frac{n}{p} + \log p)$

$\log p$ for collecting minimal distance from all processors

Proof Idea

Karlsruhe Institute of Technology

Theorem

Let $\mathbf{f_i} = |G_i|$ *with* $1 \leq i \leq k$ *be the number of points in* i*-th non-empty cell and* c, d *some positive constants, then*

$$Prob\left(\sum_{i=1}^{k} f_i^2 <= c \cdot n\right) \geq 1 - \frac{1}{2^{n^d}}$$

Proof Idea

Sampling Lemma

Let $D = \{D_1, D_2, ..., D_k\}$ be a partition of set P, $|P| = n$, for which $Pair(D) \geq n$, where

$$Pair(D) = \sum_{i=1}^{k} \frac{|D_i| \cdot (|D_i| - 1)}{2}$$

If $n^{\frac{2}{3}}$ pairwise different elements are drawn at random from P then the probability, that two elements will be chosen from the same D_i, is at least $1 - 2^{-n^c}$ for some positive constant c

Proof see

- **Rabin 1976**
- **Dietzfelbinger et al. 1997**
  Estimate the probability using Chebyshev's inequality

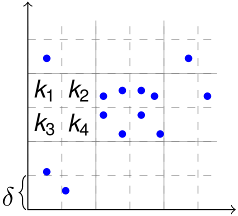
Proof Idea

Lemma 2

Let G_δ be a grid with gap δ and $G_{2\delta}$ another grid with gap 2δ, which is obtained by ignoring every second line of G_δ, then

$$N(G_{2\delta}) \leq 4N(G_\delta) + \frac{3}{2}n$$

f



Proof Idea

Lemma 3

fa

Reference I

[BS76] Jon Louis Bentley and Michael Ian Shamos. "Divide-and-conquer in multidimensional space". In: *Proceedings of the eighth annual ACM symposium on Theory of computing*. STOC '76. Hershey, Pennsylvania, USA: ACM, 1976, pp. 220–230. DOI: 10.1145/800113.803652. URL: http://doi.acm.org/10.1145/800113.803652.

[CG89] Benny Chor and Oded Goldreich. "On the power of two-point based sampling". In: *Journal of Complexity* 5.1 (1989), pp. 96–106.

[Die+97] Martin Dietzfelbinger et al. "A reliable randomized algorithm for the closest-pair problem". In: *Journal of Algorithms* 25.1 (1997), pp. 19–51.

Reference II

[FH78] Steven Fortune and John E Hopcroft. *A note on Rabin's nearest-neighbor algorithm*. Tech. rep. Cornell University, 1978.

[KM95] Samir Khuller and Yossi Matias. "A simple randomized sieve algorithm for the closest-pair problem". In: *Information and Computation* 118.1 (1995), pp. 34–37.

[KT06] J. Kleinberg and E. Tardos. "Algorithm Design". In: Pearson Education, 2006. Chap. 13 Randomized Algorithms.

[Lip09] Richard J. Lipton. *Rabin Flips a Coin*. 2009. URL: `http://rjlipton.wordpress.com/2009/03/01/rabin-flips-a-coin/` (visited on 07/15/2013).

[Rab76] Michael Oser Rabin. "Probabilistic algorithms". In: *Algorithms and Complexity: New Directions and Recent Results*. Ed. by Joseph Frederick Traub. Academic Press, 1976, pp. 21–39.

Reference III

[Weba] URL: http://www.cse.ust.hk/tcsc/comp670r/Class_1_Notes.ppt/.

[Webb] URL: http://en.wikipedia.org/wiki/Closest_pair_of_points_problem/.