PROBABILISTIC ANALYSIS OF A CLOSEST PAIR ALGORITHM

Mordecai Golin

CS-TR-194-88

December 1988

Probabilistic Analysis of a Closest Pair Algorithm

Mordecai Golin[1]
Department of Computer Science
Princeton University

December 1988

Abstract:

In this paper we give a probabilistic analysis of the recent algorithm due to Hinrichs, Nievergelt, and Schorn for solving the closest pair problem. The analysis that we present is made under the algebraic decision tree model of computation (not requiring the use of floor functions) which distinguishes it from the grid-method techniques of Bentley, Weide, and Yao. The result that we derive is that – given n points uniformly distributed in the unit square and then sorted – a simple linear sweep through the points determines the closest pair in $O(n\sqrt{\log n})$ expected time. We close the paper by discussing the possibility that it might be possible to dispense with the $\sqrt{\log n}$ factor completely; that is,the sweep algorithm might actually have a linear expected running time.

Keywords and Phrases: Computational Geometry, probabilistic analysis, closest pair problem.

1. Introduction:

The closest pair problem, that of finding the closest pair from among a set of points given in the plane, is one of the more elementary ones in computational geometry. A large number of algorithms, approaching the problem from many different perspectives, exist for its solution. There are 'practical' $O(n \log n)$ time algorithms that are relatively simple and easy to program. The divide-and-conquer method given in [BS] and the sweep-line one given in [HNS] are representative of this type. There is a 'theoretical' algorithm, [FH], that gives a slightly better running time at the expense of added complexity. There are also algorithms that solve the closest pair problem as a side effect of solving a more complicated problem such as constructing Vornoi diagrams [PS].

In this paper we shall analyze the probabilistic performance of the sweep-line algorithm given in [HNS]. The purpose is twofold; one practical, the other theoretical. The practical one is to show that, in a probabilistic sense, it is unnecessary to use complicated data structures such as 2-3 or AVL trees in the algorithm. A simple linear scan, as described in section 2, will suffice.

The theoretical purpose is subtler. Most probabilistic analyses of algorithms in computational geometry tend to fall into one of two major classes. The first class consists of those problems that are amenable to solution by grid-method techniques such as those described in [BWY]. The analyses of these algorithms takes place in a different model of computation - not the standard algebraic decision tree model. The second class of analyses tries to solve problems by applying previously known results in probabilistic geometry. A

[1] This work supported in part by NSF Grant DCR-8605962 and Office of Naval Research Grant N00014-87-K-0460.

classic example of this is the observation ([SP], pp. 145-146) that the gift-wrapping algorithm for computing convex-hulls runs in expected $O(n \log n)$ time when the points are chosen uniformly from within a convex n-gon. This follows directly from a classic theorem proven by Renyi and Sulanke that describes the distribution of the number of points on the convex hull. Unfortunately there are not all that many old results that can be directly adapted for use in this manner. To sum up: There are very few probabilistic analyses of the type so popular in the literature dealing with older, more traditional, algorithmic areas (eg. sorting) where analyses are attempted on known algorithms. Since probabilistic analyses are our only hope for predicting actual performance this leaves us with a major gap in our knowledge of the "real-life" behavior of many algorithms in computational geometry. What we attempt in this paper is such an analysis of a relatively straightforward algorithm.

2. Analysis of the Sweep Line Algorithm:

(I) Description of the algorithm:

The algorithm we consider here is the sweep line algorithm for computing the closest[1] pair. Before presenting our analysis we give a quick description of the algorithm; for a more complete description as well as a correctness proof and implementation details see [HNS].

Suppose we are given n points $p_i = (x_i, y_i), \quad i = 1 \ldots n$ in the plane. The first step is to sort them in increasing x order. The notation we use is the standard one for order statistics. The i'th sorted point is denoted by $p_{(i)} = (x_{(i)}, y_{(i)})$ where

$$x_{(1)} \le x_{(2)} \le \cdots \le x_{(n-1)} \le x_{(n)}.$$

Let δ_i be the distance between between the closest pair among the first i points, ie.

$$\delta_i = \min_{1 \le k < l \le i} ||p_{(k)} - p_{(l)}||.$$

The closest pair problem is then seen to be the problem of determining δ_n (and reporting a pair of points p, p' such that $||p - p'|| = \delta_n$. It is with the calculation of δ_n with which we will be concerned. Reporting the pair is just a matter of updating p, p' each time we decrease δ_i.)

It follows from the definition of δ_i that

$$\delta_{i+1} = \min(\delta_i, \min_{1 \le k \le i} ||p_{(i+1)} - p_{(k)}||).$$

Therefore updating δ_i is just a matter of calculating the minimum among these $||p_{(i+1)} - p_{(k)}||$. Furthermore if $\exists k \le i, \quad ||p_{(i+1)} - p_{(k)}|| < \delta_i$ then for that k, $x_{(i+1)} - x_{(k)} < \delta_i$. Thus we only have to check the distance between $p_{(i+1)}$ and those points whose x-coordinates are in the interval $(x_{(i+1)} - \delta_i, x_{(i+1)})$, a region which we will call the δ_i-interval (see figure 1(a)).

There is a geometric fact (proven in [HNS]) that is integral to Hinrichs, et al.'s implementation of their algorithm: if

$$\exists k \le i \quad \text{such that} \quad ||p_{(i+1)} - p_{(k)}|| < \delta_i$$

[1] In our description of the algorithm and in its analysis we will assume that *closest* is calculated under the standard Euclidean (L_2) Metric. This is more for ease of analysis than any other reason. We will see later that the analysis holds under any standard metric.

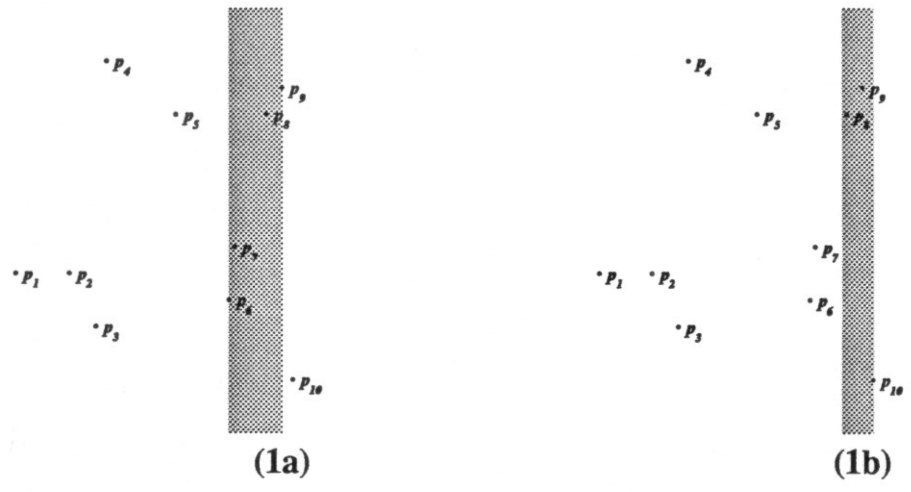

Figure 1. The minimum pair among all points preceeding p_9 is (p_1, p_2) so $\delta_8 = \|p_1 - p_2\|$. In figure 1(a) the shaded region has width δ_8 and is what we called the δ_8-interval. While checking all the points in the δ_8-interval we find that $\|p_9 - p_8\| < \delta_8$ and thus $\delta_9 = \|p_9 - p_8\|$. We now find the δ_9 interval by scanning rightward from p_6 until we hit the first point (p_8) whose x-coordinate is within δ_9 of $p_{10}.x$. This gives us figure 1(b).

---

and we sort the points in the δ_i-interval by their y-coordinates then $p_{(k)}$ has to be one of the four points immediately above, or one of the four points immediately below, the point $p_{(i+1)}$. Thus, if the points in the δ_i-interval are stored in a balanced tree sorted by y-coordinate we can calculate δ_{i+1} in $O(\log n)$ time by retrieving the eight points. All that remains is to describe how to update the δ_i-interval , ie. update the y-sorted balanced tree which contains the points in the $\delta_{(i+1)}$-*interval.* This is not difficult. First we insert $p_{(i+1)}$ into the balanced tree in $O(\log n)$ time. Then, using the fact that we already have the points sorted in increasing x-order (and have stored somewhere the index of the leftmost point in the δ_i-interval) we scan rightward deleting points from the tree until we enter the δ_{i+1}-*interval* (see fig. 1(b)). Since each point is deleted at most once during the execution of the algorithm we see that the total time for deletion is $O(n \log n)$. Furthermore, since all other operations are $O(\log n)$ time per step we find that the entire second stage of the algorithm can be implemented in $O(n \log n)$ time.

The purpose of this paper is to show that, probabilistically, the performance of the second stage of this algorithm – scanning through the sorted points while updating δ_i – will be much better than the worst case analysis suggests. In fact, we will show that it is possible to totally dispense with complicated balanced tree data structures and *still* get very good behavior. That is, if we compare $p_{(i+1)}$ to all of the points in the δ_i-interval the algorithm will still run very fast. We give a short but full listing of code for this version of the algorithm in figure 2.

More formally: given n points $p_1, \ldots p_n$ independently indentically distributed uniformly in the unit square we set

$$\mathbf{N}_i = \text{number of points in the } \delta_i\text{-interval} .$$

Figure 2. A Pascal listing of code that implements the linear scan for the closest pair. It is assumed that the points are sorted by x-coordinate in the array $p[]$ and that there is some distance function *dist*. *Left* and i point to the leftmost and rightmost points in the δ_i-interval . The **while** loop updates the points in the δ_i-interval . The inner **for** loop updates the current value of δ_i by comparing $p[i+1]$ to all of the points in the δ_i-interval .

```
left := 1;    delta := dist(p[1], p[2]);
for i := 1 to (n-1) do
    begin
        while (p[i+1].x - p[left].x) < delta
            left := left + 1;
        for j:= left to i do
            if dist(p[j], p[i+1]) < delta
                delta := dist(p[j], p[i+1]);
    end;
```

We will prove that

$$\mathrm{E}(\mathbf{N}_i) = O\left(\sqrt{\frac{n\log i}{i}}\right).$$

Since the amount of work performed while updating δ_i to δ_{i+1} is $\mathbf{N}_i$ we see that the expected total amount of work performed by the algorithm is

$$\mathrm{E}(\sum_{i=1}^{n}\mathbf{N}_i) = O\left(\sqrt{\log n}\sum_{i=1}^{n}\sqrt{\frac{n}{i}}\right) = O(n\sqrt{\log n}).$$

The remainder of this section will be devoted to proving this result.

(II) Probabilistic assumptions and notation:

We will assume that we are given n points $p_i = (x_i, y_i)$ which are I.I.D. uniformly in the unit square ($[0,1]^2$). The p_i-s are then sorted by x-coordinate and renamed $p_{(i)} = (x_{(i)}, y_{(i)})$ where

$$p_{(1)}.x \le p_{(2)}.x \le \cdots \le p_{(n-1)}.x \le p_{(n)}.x\,.$$

We need to define the following random variables:

$$\mathbf{X}_{(i)} = p_{(i)}.x$$
$$\delta_i = \min_{1\le k<l\le i} ||p_{(k)} - p_{(l)}||.$$

The number of points in an α-interval will be denoted by

$$\begin{aligned}L(X,\alpha) &= \text{ number of } p_i \text{ in the } \alpha \text{ strip to the left of } X\\ &= |\{p_j \mid X-\alpha \le p_j.x < X\}|.\end{aligned}$$

Two immediate consequences of these definitions are that

$$\forall i\ \forall \alpha, \quad L(\mathbf{X}_{(i+1)},\alpha) \le i. \tag{1}$$
$$\forall i\ \forall X\ \forall \alpha\le\beta, \quad L(X,\alpha) \le L(X,\beta). \tag{2}$$

We are now in a position to define[2] the number of points in the δ_i-interval .

$$\mathbf{N}_i = L(\mathbf{X}_{(i+1)}, \delta_i),$$

[2] A technical quibble. This definition is not valid if there are two points that have the same x-coordinate but two points sharing the same x-coordinate is a zero probability event which we can ignore.

and our goal is to analyze $\mathrm{E}(\mathbf{N}_i) = \mathrm{E}(L(\mathbf{X}_{(i+1)}, \delta_i)$ There is a major difficulty with any such analysis: If α is a constant then it is not difficult to see that

$$\mathrm{E}(L(\mathbf{X}_{(i+1)}, \alpha) \mid \mathbf{X}_{(i+1)} = x) \quad = \quad \frac{i\alpha}{x} \tag{3}$$

(where $\mathrm{E}(A|B)$ is the expected value of A given that event B occurs). This is because conditioning on $\mathbf{X}_{(i+1)} = x$ is equivalent to having the i points $p_{(1)}, \dots p_{(i)}$ be I.I.D. uniformly distributed in the rectangle

$$R = \{(s,t) \mid s \in [0,x),\ t \in [0,1]\}$$

of area x. Thus $L(\mathbf{X}_{(i+1)}, \alpha) \,|\, \mathbf{X}_{(i+1)} = x$ is a random variable with a binomial distribution on i items and parameter α/x. Its mean is therefore $i\alpha/x$.

This **does not imply** that

$$\mathrm{E}(L(\mathbf{X}_{(i+1)}, \delta_i) \,|\, \mathbf{X}_{(i+1)} = x,\ \delta_i = \alpha) \quad = \quad \frac{i\alpha}{x}$$

because conditioning on $\delta_i = \alpha$ totally changes the distribution of the points. It restricts all pair of points from being less than a distance α of each other and also requires that there is some pair that is *exactly* such a distance from each other. A good portion of our analysis will be devoted to developing a method of sidestepping this problem.

(III) Order Statistics:

Since the p_i are I.I.D. uniformly in the unit square their x-coordinates, $p_i.x$, are uniformly distributed in $[0,1]$. Thus $\mathbf{X}_{(i)}$, the sorted x-coordinates, are the order statistics of n uniformly distributed random variables on $[0,1]$. The probability density of $\mathbf{X}_{(i)}$ is well known (see [De], p. 17, for a derivation) and is given by

$$f_{(i)}(x) = \frac{n!}{(i-1)!(n-i)!}\, x^{i-1}(1-x)^{n-i} \quad x \in [0,1].$$

Using the standard properties of the Beta and Gamma functions we can immediately calculate all of the moments of $\mathbf{X}_{(i)}$ as in the following lemma:

Lemma 1.

$$\mathrm{E}(\mathbf{X}_{(i)}^k) = \frac{n!\,(k+i-1)!}{(i-1)!\,(n+k)!}.$$

Proof.

$$\begin{aligned}\mathrm{E}(\mathbf{X}_{(i)}^k) &= \frac{n!}{(i-1)!(n-i)!} \int_0^1 x^{k+i-1}(1-x)^{n-i}\, dx \\ &= \frac{n!}{(i-1)!(n-i)!} \beta(k+i, n-i+1) \\ &= \frac{n!\,(k+i-1)!}{(i-1)!\,(n+k)!}.\end{aligned}$$

This has the immediate corollary:

Corollary 1:

$$\begin{aligned}\mathrm{E}(\mathbf{X}_{(i)}) &= \frac{i}{n+1}. \\ \mathrm{E}(\mathbf{X}_{(i)}^2) &= \frac{i\,(i+1)}{(n+1)\,(n+2)}. \\ \mathrm{Var}(\mathbf{X}_{(i)}) = \mathrm{E}(\mathbf{X}_{(i)}^2) - \mathrm{E}^2(\mathbf{X}_{(i)}) &= \frac{i\,(n+1-i)}{(n+2)\,(n+1)^2}.\end{aligned}$$

This in turn leads us directly to

Corollary 2:

$$\Pr\left(\mathbf{X}_{(i+1)} \notin \left[\tfrac{1}{2}\tfrac{(i+1)}{(n+1)}, \tfrac{3}{2}\tfrac{(i+1)}{(n+1)}\right]\right) \le \frac{4}{i+1}. \tag{4}$$

Proof. By Chebyshov's inequality and Corollary 1

$$\begin{aligned}\Pr\left(\mathbf{X}_{(i+1)} \notin \left[\tfrac{1}{2}\tfrac{(i+1)}{(n+1)}, \tfrac{3}{2}\tfrac{(i+1)}{(n+1)}\right]\right) &= \Pr\left(|\mathbf{X}_{(i+1)} - \mathbf{E}(\mathbf{X}_{(i+1)})| > \frac{(i+1)}{2(n+1)}\right)\\ &\le \mathrm{Var}(\mathbf{X}_{(i+1)})\left(\frac{2(n+1)}{(i+1)}\right)^2\\ &\le \frac{4}{(i+1)}.\end{aligned}$$

(IV) Problem formulation and analysis:

In this section we will derive asymptotics for $\mathbf{E}(\mathbf{N}_i) = \mathbf{E}(L(\mathbf{X}_{(i+1)}, \delta_i))$. We will prove

Theorem 1:

$$\mathbf{E}(\mathbf{N}_i) = \mathbf{E}(L(\mathbf{X}_{(i+1)}, \delta_i)) = O\left(\sqrt{\frac{n\log i}{i}}\right). \tag{5}$$

Proof: The proof is complex and will be split into three parts. In the first we will show that we can effectively restrict $\mathbf{X}_{(i+1)}$ to a bounded subinterval $D_i = \left[\tfrac{1}{2}\tfrac{(i+1)}{(n+1)}, \tfrac{3}{2}\tfrac{(i+1)}{(n+1)}\right]$. In the second we will use this restriction to D_i to bound $\mathbf{E}(L(\mathbf{X}_{(i+1)}, \delta_i)\,|\,\mathbf{X}_{(i+1)} = x)$. Finally, in the third, we will combine the results obtained in the first two parts to prove the theorem.

(i) Writing out the standard formula for conditional expectation (see, for example, [GS]) we find that

$$\begin{aligned}\mathbf{E}(L(\mathbf{X}_{(i+1)}, \delta_i)) &= \int_0^1 \mathbf{E}\left(L(\mathbf{X}_{(i+1)}, \delta_i)\,|\,\mathbf{X}_{(i+1)} = x\right) f_{(i+1)}(x)\,dx\\ &\le \int_{x\in D_i} \mathbf{E}\left(L(\mathbf{X}_{(i+1)}, \delta_i)\,|\,\mathbf{X}_{(i+1)} = x\right) f_{(i+1)}(x)\,dx\\ &\qquad + \Pr\left(\mathbf{X}_{(i+1)} \notin D_i\right)\cdot \max_{x\in[0,1]} L(x, \delta_i).\end{aligned} \tag{6}$$

From (1) and (4) we see that

$$\Pr\left(\mathbf{X}_{(i+1)} \notin D_i\right)\cdot \max_{x\in[0,1]} L(x, \delta_i) \le \frac{4i}{i+1} < 4.$$

We can therefore rewrite (6) as

$$\mathbf{E}(L(\mathbf{X}_{(i+1)}, \delta_i)) \le 4 + \max_{x\in D_i} \mathbf{E}\big(L(\mathbf{X}_{(i+1)}, \delta_i)\,|\,\mathbf{X}_{(i+1)} = x\big). \tag{7}$$

(ii) Our approach will be to bound (7) by a judicious choice of α in the equality

$$\begin{aligned}\mathbf{E}\left(L(\mathbf{X}_{(i+1)}, \delta_i)\,|\,\mathbf{X}_{(i+1)} = x\right) &= \mathbf{E}\big(L(\mathbf{X}_{(i+1)}, \delta_i)\,|\,\mathbf{X}_{(i+1)} = x,\ \delta_i \le \alpha\big)\cdot\Pr\big(\delta_i \le \alpha\,|\,\mathbf{X}_{(i+1)} = x\big)\\ &\quad + \mathbf{E}\big(L(\mathbf{X}_{(i+1)}, \delta_i)\,|\,\mathbf{X}_{(i+1)} = x,\ \delta_i > \alpha\big)\cdot\Pr\big(\delta_i > \alpha\,|\,\mathbf{X}_{(i+1)} = x\big).\end{aligned} \tag{8}$$

Thus we must bound $\Pr(\delta_i > \alpha\,|\,\mathbf{X}_{(i+1)} = x)$. We can do this by recalling that conditioning on $\mathbf{X}_{(i+1)} = x$ is equivalent $p_{(1)}, \ldots, p_{(i)}$ being uniformly distributed in the rectangle

$$R = \{(s,t)\,|\,s \in [0,x),\ t\in[0,1]\}.$$

What follows is just a standard area argument: We define B_k as the $\alpha/2$ ball around $p_{(k)}$

$$B_k = B\left(p_{(k)}, \frac{\alpha}{2}\right) = \left\{q \, : \, \|q - p_{(k)}\| < \frac{\alpha}{2}\right\}$$

we see that

$$\delta_i > \alpha \quad \Leftrightarrow \quad B_k \cap B_l = \emptyset, \qquad \forall l, \, k : \, 1 \le k < l \le i.$$

Since the $p_{(k)}$ are I.I.D. uniformly in R we find that

$$\Pr(\delta_i > \alpha \,|\, \mathbf{X}_{(i+1)} = x) \le \prod_{k=2}^{i} \Pr\Big(p_{(k)} \notin \bigcup_{1 \le j < k} B_l \,\Big|\, \quad B_l \cap B_m = \emptyset, \quad {\scriptstyle 1 \le l < m < k}\Big). \tag{9}$$

Since R is a rectangle and $B_l \in R$ we see that $B_l \cap R$ includes at least a quarter of the area of B_l so

$$area\,(B_l \cap R) \le \frac{\pi\alpha^2}{16}.$$

This, together with the fact that $area(R) = x$, gives

$$\Pr\Big(p_{(k)} \notin \bigcup_{1 \le j < k} B_l \,\Big|\, \quad B_l \cap B_m = \emptyset, \, {\scriptstyle 1 \le l < m < k}\Big) \le 1 - \frac{(k-1)\pi\alpha^2}{16x}.$$

Plugging this back into (9) yields

$$\Pr(\delta_i > \alpha \,|\, \mathbf{X}_{(i+1)} = x) \le \prod_{j=1}^{i-1} \left(1 - \frac{j\pi\alpha^2}{16x}\right). \tag{10}$$

Since $x \in \left[\frac{1}{2}\frac{(i+1)}{(n+1)}, \frac{3}{2}\frac{(i+1)}{(n+1)}\right]$ we have $x < 2i/n$ (for $i \ge 3$) and thus

$$\Pr(\delta_i > \alpha \,|\, \mathbf{X}_{(i+1)} = x) \le \prod_{j=1}^{i-1} \left(1 - \frac{jc\alpha^2 n}{i}\right) \tag{11}$$

where $c = \pi/32$. Taking logarithms and using $\ln(1-x) = -x + O(x^2)$ we find

$$\begin{aligned} \Pr(\delta_i > \alpha \,|\, \mathbf{X}_{(i+1)} = x) \le& \exp\left(\sum_{j<i} \ln\left(1 - \frac{cjn\alpha^2}{i}\right)\right) \\ =& \exp\left(-\frac{i(i-1)cn\alpha^2}{i} + O(in^2\alpha^4)\right) \\ =& e^{-icn\alpha^2}\left(1 + O(in^2\alpha^4 + n\alpha^2)\right). \end{aligned} \tag{12}$$

Setting $\alpha' = \sqrt{\frac{2}{c}\frac{\ln i}{in}}$ yields

$$\begin{aligned} \Pr\left(\delta_i > \alpha' \,|\, \mathbf{X}_{(i+1)} = x\right) \le& e^{-2\ln i}\left(1 + O\left(\frac{\ln^2 i}{i}\right)\right) \\ =& O\left(\frac{1}{i^2}\right). \end{aligned} \tag{13}$$

We combine this result with (1) to bound the second term on the right hand side of (8):

$$\mathbf{E}\left(L(\mathbf{X}_{(i+1)}, \delta_i) \,|\, \mathbf{X}_{(i+1)} = x, \, \delta_i > \alpha'\right) \cdot \Pr\left(\delta_i > \alpha' \,|\, \mathbf{X}_{(i+1)} = x\right) \le i\, O\left(\frac{1}{i^2}\right) = O\left(\frac{1}{i}\right). \tag{14}$$

All that is left is to bound the first term of (8). As a first step notice that if $\delta_i \leq \alpha'$ we can use (2) to derive

$$\mathbf{E}\left(L(\mathbf{X}_{(i+1)}, \delta_i) \,|\, \mathbf{X}_{(i+1)} = x,\ \delta_i \leq \alpha'\right) \leq \mathbf{E}\left(L(\mathbf{X}_{(i+1)}, \alpha') \,|\, \mathbf{X}_{(i+1)} = x,\ \delta_i \leq \alpha'\right). \tag{15}$$

Since $L(\mathbf{X}_{(i+1)}, \alpha')$ is a a nonnegative random variable

$$\mathbf{E}(L(\mathbf{X}_{(i+1)}, \alpha') \,|\, \mathbf{X}_{(i+1)} = x,\ \delta_i \leq \alpha') \cdot \Pr(\delta_i \leq \alpha' \,|\, \mathbf{X}_{(i+1)} = x) \leq \mathbf{E}(L(\mathbf{X}_{(i+1)}, \alpha') \,|\, \mathbf{X}_{(i+1)} = x). \tag{16}$$

From (3) and the fact that $x \geq \frac{i}{2n}$ we know that

$$\mathbf{E}(L(\mathbf{X}_{(i+1)}, \alpha') \,|\, \mathbf{X}_{(i+1)} = x) = \frac{i\alpha'}{x} \leq \sqrt{\frac{\frac{8}{c}\, n \ln i}{i}}. \tag{17}$$

Equations (15), (16), and (17) taken together imply that

$$\mathbf{E}(L(\mathbf{X}_{(i+1)}, \delta_i) \,|\, \mathbf{X}_{(i+1)} = x,\ \delta_i \leq \alpha') \cdot \Pr(\delta_i \leq \alpha' \,|\, \mathbf{X}_{(i+1)} = x) = O\left(\sqrt{\frac{n \ln i}{i}}\right). \tag{18}$$

We conclude this section by noting that (14) and (18) are just the two terms on the right hand side of (8) with $\alpha = \alpha'$ so

$$\mathbf{E}\left(L(\mathbf{X}_{(i+1)}, \delta_i) \,|\, \mathbf{X}_{(i+1)} = x\right) = O\left(\sqrt{\frac{n \ln i}{i}}\right). \tag{19}$$

(iii) We are now in a position to finish the proof of theorem 1. In (i) we proved that

$$\mathbf{E}(L(\mathbf{X}_{(i+1)}, \delta_i)) \leq 4 + \max_{x \in D_i} \mathbf{E}\big(L(\mathbf{X}_{(i+1)}, \delta_i) \,|\, \mathbf{X}_{(i+1)} = x\big) \tag{7}$$

where $D_i = \left[\frac{1}{2}\frac{(i+1)}{(n+1)}, \frac{3}{2}\frac{(i+1)}{(n+1)}\right]$. In (ii) we showed that, for $x \in D_i$,

$$\mathbf{E}\left(L(\mathbf{X}_{(i+1)}, \delta_i) \,|\, \mathbf{X}_{(i+1)} = x\right) = O\left(\sqrt{\frac{n \ln i}{i}}\right) \tag{19}$$

where the constant implicit in the $O()$ notation is independent of i and n. Thus

$$\max_{x \in D_i} \mathbf{E}\big(L(\mathbf{X}_{(i+1)}, \delta_i) \,|\, \mathbf{X}_{(i+1)} = x\big) = O\left(\sqrt{\frac{n \ln i}{i}}\right)$$

and we are done.

Q.E.D.

Conclusions, simulation results and directions for future research:

We proved that – given n points uniformly distributed in the unit square and then sorted – a simple linear scan through the points of the type described in section 2(I) will determine the closest pair in expected $O(n\sqrt{\log n})$ time. In our analysis we assumed that distance was defined by the L_2 metric but a few simple observations show that the analysis continues to be valid under the L_∞ and L_p metrics. First, notice that the analysis of order statistics given in section 2(III) is valid under the L_∞ and any L_p metrics because it takes place in a one-dimensional space. Next, notice that the only two properties of L_2 that we used in 2(IV)

n	$\sum_i \mathbf{N}_i$
500	688.7
600	820.4
700	909.7
800	1039.8
900	1209.5
1000	1303.3
1500	1912.4
2000	2815.5
2500	3352.3
3000	3898.9

Table 1. For each value of n 100 random sets of n random points were generated. Each set was sorted and $\sum \mathbf{N}_i$ was calculated by running the algorithm of figure 2. The value in the second column is the average value of $\sum \mathbf{N}_i$ over all 100 point sets.

---

were that $||p_{(k)} - p_{(i+1)}|| < \delta_i$ implies that $p_{(k)}$ is in the δ_i-interval , and that equation (3) is correct. Both of these properties are invarient under a switch to the L_∞ or L_p metrics so the analysis remains the correct.

An interesting open question is whether or not it is possible to improve the result obtained in Theorem 1. It is not difficult to derive from equation (12) that

$$\mathbf{E}\left(\delta_i \,|\, \mathbf{X}_{(i+1)} = x\right) = O\left(\frac{1}{\sqrt{in}}\right).$$

If δ_i were *concentrated* at its mean then we could modify the analysis leading to equation (18) to yield

$$\mathbf{E}(\mathbf{N}_i) = \mathbf{E}(L(\mathbf{X}_{(i+1)}, \delta_i)) = O\left(\sqrt{\frac{n}{i}}\right).$$

This would give the second stage of the algorithm expected linear time and in fact, simulations run on random points do seem to show linear time behavior. Unfortunately, the complications (discussed in section 2(II)) that are introduced by conditioning on $\delta_i > \alpha$ seem to preclude an extension of our analysis in that direction.

Finally, we ran the algorithm of figure 2 on random point sets. For each value of n in table 1 we ran the algorithm on 100 sets of n random points and averaged the observed values of $\sum \mathbf{N}_i$. Our results show good performance but they can not provide us with any evidence that the expected running time of the algorithm is actually linear; for n small enough for simulations to be practical $\sqrt{\log n}$ is too small to distinguish from a constant.

REFERENCES:

[BS] J.L. Bentley and M.I. Shamos, "Divide-And-Conquer in Multidimensional Space," *Proc. 8th Annual Symp. on the Theory of Computing,* (1976) 220-230.

[BWY] J.L. Bentley, B.W. Weide, and A.C. Yao, "Optimal Expected -Time Algorithms for Closest Point Problems," *ACM Transactions on Mathematical Software,* **6**(4) (1980) 563-580.

[De] Luc Devroye, *Non-Uniform Random Variate Generation,* Springer-Verlag, New York. (1986).

[FH] S. Fortune and J.E. Hopcroft, "A Note on Rabin's Nearest Neighbor Algorithm," *Information Processing Letters* **8** (1) (1979) 20-23.

[GS] G.R. Grimmet and D. Stirzaker, *Probability and Random Processes,* Clarendon Press, Oxford. (1985).

[HNS] K. Hinrichs, J. Nievergelt, and P. Schorn, "Plane-Sweep Solves the Closest Pair Problem Elegantly," *Information Processing Letters* **26** (11 Jan. 1988) 255-261.

[PS] F.P. Preparata and M.I. Shamos, *Computational Geometry: An Introduction,* Springer-Verlag, New York. (1985).