60

第35卷 第10期 计算机研究与发展 Vol. 35, No. 10
1998年10月 COMPUTER RESEARCH & DEVELOPMENT Oct. 1998

求平面点集最近点对的一个改进算法

956-960 TP301.6

O24

周玉林 熊鹏荣 朱 洪*
(上饶师范专科学校数学系 上饶 334001)
*(复旦大学计算机科学系 上海 200433)

摘 要 文中对Preparata和Shamos在1985年提出的求平面点集最近点对的一个分治算法进行了改进，使原来归并时最多需计算 $3n$ 对点对的距离，改进为最多只需计算 $2n$ 对点对的距离，计算距离的复杂度在最坏的情况下由原来的 $3n\log n$ 减少到现在的 $2n\log n$.

关键词 分治算法，最近点对，算法复杂性 计算几何 平面点集

中图法分类号 TP301.6

AN IMPROVED ALGORITHM ABOUT THE CLOSEST PAIR OF POINTS ON PLANE SET

Zhou Yulin, Xiong Pengrong, and Zhu Hong*
(*Department of Mathematics, Shangrao Teachers College, Shangrao* 334001)
* (*Department of Computer Science, Fudan University, Shanghai* 200433)

Abstract In the paper the divide-and-conquer algorithm about the closest pair of points on plane set is improved, which was pressented by Preparata and Shamos in 1985. Their algorithm needs at most $3n$ calculations on distance, and the time complexity is $3n\log n$ in worst case. The improved algorithm only needs at most $2n$ calculations on distance, and the time complexity of calculation on distance is reduced to $2n\log n$.

Key words divide-and-conquer algorithm, the closest pair of points, algorithmic complexity

Class number TP301.6

1 引 言

求平面点集最近点对问题，是计算几何中的一个基本的重要问题，该问题在交通控制系统等方面都有广泛的应用.

已知平面点集 Q，$|Q|=n$，Q 上共可构成 $C_n^2=\frac{n(n-1)}{2}$ 对点对，求 Q 中最近点对(欧氏距离)的最简单朴素的办法是对这 $\frac{n(n-1)}{2}$ 对点对逐一计算其距离，由此找出距离最小者，这种方法所需时间的代价为 $O(n^2)$.

原稿收到日期：1997-04-01；修改稿收到日期：1997-08-20. 本课题得到国家自然科学基金资助. 周玉林，讲师，主要研究方向为算法理论. 熊鹏荣，主要研究方向为算法理论. 朱洪，教授，主要研究方向为计算理论、算法理论及程序理论等.

1985 年 Preparata 和 Shamos[1]给出了该问题的一个分治算法（该算法在文献[2]、[3]等中都有介绍和讨论），该算法在分治过程中每次选择一条垂直于 x 轴的直线 l，将平面点集平均分成 Q_L，Q_R 左右两部分，依次找出左、右两部分点集的最近点对的距离，分别记为 δ_L，δ_R，记 $\delta=\min(\delta_L,\delta_R)$，然后在合并时对分界线左右宽为 2δ 的垂直的带形区域上的跨越左右两集的点对进行检验，以检验原点集的最近点对是否可能在该带域中，如图 1.

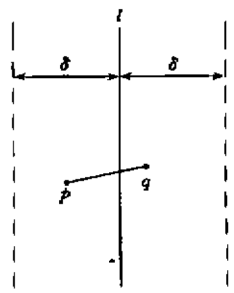

图 1

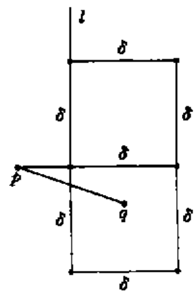

图 2

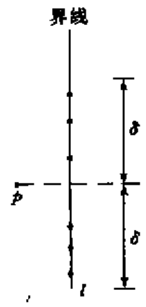

图 3

Preparata 和 Shamos 分析了最近点对可能在这样的带域中出现的情况，若(p,q)是 Q 的最近点对，p 在带域左半部分，则 q 点必在图 2 中所示的 $\delta\times2\delta$ 长方形上，而在该长方形上，最多只能有右边点集的 6 个点. 因此，在考察带域的最近点对时，只要对带域左半部分的每个点 p，检验带域右半部分其纵坐标在 p 点纵坐标周围（$\pm\delta$ 范围内）的最多 6 个点（上下各 3 个），如图 3. 因此该算法在最坏情况下（此时，Q 的点全在宽为 2δ 的带域上），在合并时需计算 $3n$ 对点对的距离.

2 对 Preparata-Shamos 算法的改进

Preparata-Shamos 算法在考察带域是否存在最近点对时，对带型左半区域中的每个点 p，要检验相对应的右半区域中的至多 6 个点，即至多要考察 6 对点对的距离，我们通过观察发现，对点 p 仅需最多只检验右半带域中其 y 坐标与 p 最近的最多 4 个点即可（上下各 2 点见图 4）.

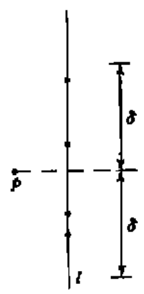

图 4

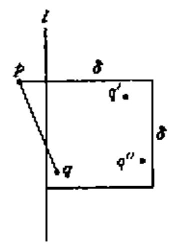

图 5

定理 1. 如图 5 所示，设 $q(x_q,y_q)$是 Q_R 中在以 δ 为边的正方形上与点 $p(x_p,y_p)$距离 $\delta_1=d(p,q)<\delta$ 的点，则在该正方形上最多还有 2 个 Q_R 中点 $q'(x_{q'},y_{q'})$，$q''(x_{q''},y_{q''})$满足：

$$y_q\leqslant y_{q'},y_{q''}\leqslant y_p,$$

而且当有两个点时，q' 与 q''中必有一个点与 p 的距离小于等于 δ_1.

证明. 将正方形坐标大于等于 y_q 的上半部分分成三部分，如图6所示.

第1部分，$MTADEF$，这一部分内的点和 q 的距离小于 δ，故不会含有 Q_R 中的点；

第2部分，CEF，这一部分内两点间的距离 $\leqslant EF < pE = \delta_1 < \delta$，故最多只有 Q_R 中的一个点 q'，而且如果有的话，它和 p 的距离小于等于 δ_1；

第3部分，$DBCE$，这一部分内的两点间距离 $\leqslant CD \leqslant CA$，而

$$\begin{aligned} CA^2 &= BC^2 + AB^2 = (\delta - (\delta_1 - \varepsilon))^2 + TM^2 \\ &\leqslant \delta^2 + \delta_1^2 + \varepsilon^2 - 2\delta\delta_1 - 2\delta_1\varepsilon + 2\delta\varepsilon + \delta_1^2 - \varepsilon^2 \\ &= \delta^2 - 2(\delta - \delta_1)(\delta_1 - \varepsilon) < \delta^2 \end{aligned}$$

所以，这一部分内两点间的距离小于等于 δ，即至多只有一个点 q''. 证毕.

定理2. 在检验上述带域(图1)是否存在小于 δ 的跨域最近点对时，只要对其左半域上的每一点 p，检验右半域 y 坐标与 p 最近的至多4个点即可(上下各2个，见图4).

证明. 由定理1及 p 点右半区域的右上、右下正方形的对称性，在检验时，对左半域上的每一点 p，最多只要验证其相应的右半域中相应的上、下正方形中与该点的 y 坐标最接近的各2个点，就一定不会漏过对最近点对的检验. 证毕.

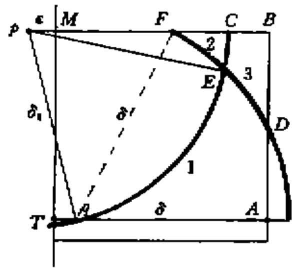

图6

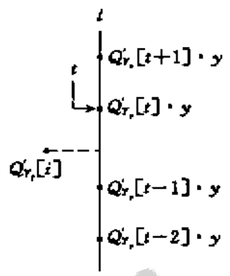

图7

下面我们给出求平面点集的最近点对的算法.

下述集合 Q，数组 Q_X，Q_Y 等其元素都以基本数据结构

```
node=record
    x,y:real;
  end
```

为基础.

算法.

1 对 Q 预排序，先按 y 增序排序得数组 Q_Y，再对 Q_Y 按 x 的增序(用稳定排序法)排序得数组 Q_X.

2 $\delta \leftarrow \infty$；

3 $DIVIDE\text{-}CONQUER$ (Q_X, Q_Y, δ)(分治)；

3.1 if $|Q_X| \leqslant 2$ then

  if $|Q_X| = 2$ and $d(Q_X[1], Q_X[2]) < \delta$ then $\delta \leftarrow d(Q_X[1], Q_X[2])$；

  return (δ)；

3.2 $j_1 \leftarrow 1$; $j_2 \leftarrow 1$; $k_1 \leftarrow 1$; $k_2 \leftarrow 1$; $mid = \lfloor |Q_X|/2 \rfloor$; $l \leftarrow Q_X[mid].x$;

3.3 for $i \leftarrow 1$ to $|Q_X|$ do

3.3.1 if $i \leqslant mid$ then $Q_{X_L}[k_1] \leftarrow Q_X[i]$, $k_1 \leftarrow k_1 + 1$

  else $Q_{X_R}[k_2] \leftarrow Q_X[i]$; $k_2 \leftarrow k_2 + 1$(将数组 Q_X 对半分成数组 Q_{X_L}, Q_{X_R})

3.3.2 if $Q_Y[i].x < l$ or $(Q_Y[i].x = l$ and $Q_Y[i].y \leqslant Q_X[mid].y)$

then $Q_{Y_L}[j_1] \leftarrow Q_Y[i]$; $j_1 \leftarrow j_1+1$

else $Q_{Y_R}[j_2] \leftarrow Q_Y[i]$; $j_2 \leftarrow j_2+1$ {将数组 Q_Y 平均分成数组 Q_{Y_L}, Q_{Y_R}}

3.4 $\delta_L \leftarrow DIVIDE\text{-}CONQUER\ (Q_{X_L}, Q_{Y_L}, \delta)$;

3.5 $\delta_R \leftarrow DIVIDE\text{-}CONQUER\ (Q_{X_R}, Q_{Y_R}, \delta)$;

3.6 $\delta' \leftarrow \min(\delta_L, \delta_R)$;

3.7 $\delta \leftarrow MERGE\ (Q_{Y_L}, Q_{Y_R}, \delta')$;

3.8 return(δ).

过程 $MERGE(Q_{Y_L}, Q_{Y_R}, \delta')$

(1) $i_1 \leftarrow 1$; $i_2 \leftarrow 1$;

(2) for $i \leftarrow 1$ to $|Q_{Y_L}|$ do

if $Q_{Y_L}[i].x > l-\delta'$ then $Q'_{Y_L}[i_1] \leftarrow Q_{Y_L}[i]$; $i_1 \leftarrow i_1+1$;

(3) for $i \leftarrow 1$ to $|Q_{Y_R}|$ do

if $Q_{Y_R}[i].x < l+\delta'$ then $Q'_{Y_R}[i_2] \leftarrow Q_{Y_R}[i]$; $i_2 \leftarrow i_2+1$;

(4) $t \leftarrow 1$;

(5) for $i \leftarrow 1$ to $|Q'_{Y_L}|$ do

① while $Q'_{Y_R}[t].y < Q'_{Y_L}[i].y$ and $t < |Q'_{Y_R}|$ do

$t \leftarrow 1+1$;

② for $j \leftarrow \max(t-2, 1)$ to $\min(t+1, |Q'_{Y_R}|)$ do

if $(Q'_{Y_L}[i].y-\delta' < Q'_{Y_R}[j].y < Q'_{Y_L}[i].y+\delta')$ and $(Q'_{Y_R}[j].x < l+\delta')$

and $(d(Q'_{Y_L}[i], Q'_{Y_R}[j]) < \delta')$

then $\delta' \leftarrow d(Q'_{Y_L}[i], Q'_{Y_R}[j])$;

(6) return (δ');

上述算法的分治算法中，3.1 先判断集合元素数是否小于等于 2，若小于等于 2，则返回距离；3.2 给出了平面点集划分成左、右区域的分界线($x=l$)，该分界线以数组 Q_X 的最中间的一个元素的 x 值为分界值；3.3.1 将数组 Q_X 对半切成两部分；3.3.2 将 Q_Y 分成左、右两部分，$Q_{Y_L}=\{p \in Q_Y | p \in Q_{X_L}\}$，$Q_{Y_R}=\{p \in Q_Y | p \in Q_{X_R}\}$，注意到：

$$p \in Q_{Y_L} \Longleftrightarrow x_p < Q_X[mid].x \text{ 或 } x_p = Q_X[mid].x \text{ 且 } y_p \leqslant Q_X[mid].y.$$

这样，平均地将数组(点集)分成两部分，在分界线 l 上的点按界点 $y=Q_X[mid].y$ 划分，界点下的部分的点(包括界点)属于 Q_{Y_L}，界点上的部分的点属于 Q_{Y_R}. 以上划分得到的数组都是保持原序的，且 Q_{X_L} 中的点与 Q_{Y_L} 中所含的点相同，Q_{X_R} 中的点与 Q_{Y_R} 中所含的点相同. 3.4, 3.5 递归调用，3.7 合并.

在合并过程中，第(2)，(3)两步滤去带形区域以外的点，得相应的数组 Q'_{Y_L} 和 Q'_{Y_R}，第(4)步中的 t 是一指针，该指针每次定位于与点 $Q'_{Y_L}[i]$ 对应的 Q'_{Y_R} 中其纵坐标大于等于 $Q'_{Y_L}[i].y$ 且纵坐标与 $Q'_{Y_L}[i].y$ 距离最近的点(由①实现)，见图 7. 第(5)步的循环每当发现一更近的点对时，就立即缩小 δ' 的值，这样相应地缩小了受检的长方形区域，②检验 $\delta' * 2\delta'$ 上的最多 4 个点.

3 算法分析

比较 Preparata-Shamos 算法和我们的算法，我们略去算法细节上的差别，假设 Shamos，Preparata 算法及我们的算法的预排序的时间代价都是 $2n\log n$，两算法的主要差别是在分治算法的归并中，Preparata-Shamos 算法对 Q_{Y_L} 中的每个点，都要检验 Q_{Y_R} 中的 6 个点，而我们的算法只要检验 4 个点，于是，Preparata-Shamos 算法在分治算法中对距离的计算的时间代价的递归式为 $T(n)=2T(n/2)+(n/2)\times 6$，其解为 $T(n) \leqslant 3n\log n$，而我们的算法的对距离的计算的时间代价的递归式为 $T(n)=2T(n/2)+(n/2)\times 4$，其解为 $T(n)$

$\leq 2n\log n$，因为平面点集最近点对问题算法的时间开销最主要是在对距离的计算上，所以，我们算法的时间复杂度 $2n\log n$，比 Preparata-Shamos 的 $3n\log n$ 低.

4 问 题

对空间点集的最近点对问题，只要对平面点集最近点对的分治算法稍作修改即可适合求空间点集的最近点对问题，设 Q 是空间点集，对 Q 按 x 坐标预排序得数组 Q_X，再对 Q_X 先按 y 后按 z 排序得数组 W. 分治时用垂直 x 轴的平面将空间平均分为两部分，相应地数组平均分成 2 组，归并时，对该平面两侧的 δ 宽的区域的上侧的每个点，最多只检验其下侧对应的 $2\delta\times2\delta\times\delta$ 的长方体区域的最多 18 个点，故其分治算法的时间复杂度最多是 $9n\log n$. 本文提出，在空间情况下，是否能将每次考察 18 个点降到 12 个点？或最多可降到仅考虑多少个点？仍是一个值得讨论的问题.

参 考 文 献

1 Preparata F P, Shamos M I. Computational Geometry: An Introduction. New York: Springer-Verlag, 1985

2 Cormen T H, Leiserson C E, Rivest R L. Introduction to Algorithms. New York: McGraw-Hill, 1993

3 Zhu Hong *et al.* The Design and Analysis of Algorithms (in Chinese). Shanghai: Shanghai Publishing House on Literature of Science and Technology, 1989

(朱洪等. 算法设计和分析. 上海：上海科技文献出版社，1989)