$\mathcal{O}(n)$ 复杂度的随机最近点对算法[1]

wzy

SUSTC

April 2, 2025

Content

记号及约定

- $\delta(S)$ 表示二维点集 S 中的最近点对的距离,
- 对于 $x \in S$, $d(x)$ 表示 x 到 S 中的其他点的最近距离,
- S_i 表示第 i 轮迭代时的点集。

筛选过程

算法的核心思想是通过筛选法求出 $\delta(S)$ 的一个近似解，然后再通过这个近似解求出 $\delta(S)$。筛选过程如下：

筛选过程

算法的核心思想是通过筛选法求出 $\delta(S)$ 的一个近似解，然后再通过这个近似解求出 $\delta(S)$。筛选过程如下：

初始化 $S_1 \leftarrow S$，随机选取一个点 $x_i \in S_i$ 并求出 $d(x_i)$（此时 d 是相对于 S_i 的，后文同理）。

筛选过程

算法的核心思想是通过筛选法求出 $\delta(S)$ 的一个近似解，然后再通过这个近似解求出 $\delta(S)$。筛选过程如下：

初始化 $S_1 \leftarrow S$，随机选取一个点 $x_i \in S_i$ 并求出 $d(x_i)$（此时 d 是相对于 S_i 的，后文同理）。

将平面按照 $b = d(x_i)/3$ 分块，若一个点的 *8-邻居* 块都为空且其自身所在的块仅有他自己，则删去这个点。删去所有符合条件的点后，得到 S_{i+1}。

筛选过程

算法的核心思想是通过筛选法求出 $\delta(S)$ 的一个近似解，然后再通过这个近似解求出 $\delta(S)$。筛选过程如下：

初始化 $S_1 \leftarrow S$，随机选取一个点 $x_i \in S_i$ 并求出 $d(x_i)$（此时 d 是相对于 S_i 的，后文同理）。

将平面按照 $b = d(x_i)/3$ 分块，若一个点的 *8-邻居* 块都为空且其自身所在的块仅有他自己，则删去这个点。删去所有符合条件的点后，得到 S_{i+1}。

不断进行上述过程直到将点集删空（每轮迭代至少删除一个点），记最小的使得 $S_{i^*+1} = \varnothing$ 的时刻为 i^*。

筛选过程

引理 1

$d(x_{i^*})$ 满足 $\delta(S) \le d(x_{i^*}) < 3\delta(S)$。

筛选过程

引理 1

$d(x_{i^*})$ 满足 $\delta(S) \le d(x_{i^*}) < 3\delta(S)$。

证明：左半边是平凡的，考虑证明右半边。首先有两个观察：

筛选过程

引理 1

$d(x_{i^*})$ 满足 $\delta(S) \le d(x_{i^*}) < 3\delta(S)$。

证明：左半边是平凡的，考虑证明右半边。首先有两个观察：

1. 满足 $d(x) > 2\sqrt{2}b$ 的点一定会被删除，
2. 满足 $d(x) < b$ 的点一定会被保留。

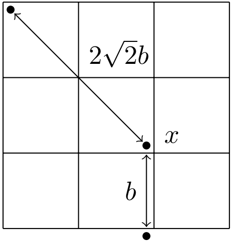
筛选过程

引理 1

$d(x_{i^*})$ 满足 $\delta(S) \le d(x_{i^*}) < 3\delta(S)$。

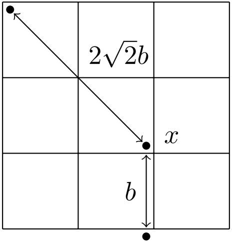
证明：左半边是平凡的，考虑证明右半边。首先有两个观察：

1. 满足 $d(x) > 2\sqrt{2}b$ 的点一定会被删除，
2. 满足 $d(x) < b$ 的点一定会被保留。



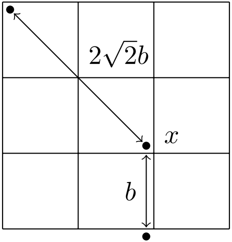
筛选过程

如图，方格中能塞下的最长距离为 $2\sqrt{2}b$，$b < d(x) < 2\sqrt{2}b$ 的点可能被保留也可能被删除。由于 $2\sqrt{2}/3 < 1$，所以 $d(x) \geq d(x_i)$ 的 x 都被删除了，因此每轮迭代中的 $d(x_i)$ 是**递减**的。



筛选过程

如图，方格中能塞下的最长距离为 $2\sqrt{2}b$，$b < d(x) < 2\sqrt{2}b$ 的点可能被保留也可能被删除。由于 $2\sqrt{2}/3 < 1$，所以 $d(x) \geq d(x_i)$ 的 x 都被删除了，因此每轮迭代中的 $d(x_i)$ 是**递减**的。



令 (u, v) 为 S 中的最近点对，j^* 表示 (u, v) 任一点被删除的最早时间，根据上面的性质我们知道 $\delta(S) \geq d(x_{j^*})/3 > d(x_{i^*})/3$，即 $d(x_{i^*}) < 3\delta(S)$。 □

求解最近点对

此时我们已经得到了一个 $\delta(S)$ 的近似解 $d(x_{i^*})$。再将平面按照 $b = d(x_{i^*})$ 分块，由于块长是 $\delta(S)$ 的常数倍，因此每个块内只有 $\mathcal{O}(1)$ 个点。

求解最近点对

此时我们已经得到了一个 $\delta(S)$ 的近似解 $d(x_{i^*})$。再将平面按照 $b = d(x_{i^*})$ 分块，由于块长是 $\delta(S)$ 的常数倍，因此每个块内只有 $\mathcal{O}(1)$ 个点。

对于每个点，检查其所在的块以及其 *8*-邻居 块，用这些块内的点更新答案（其余的块距离该点 $> \delta(S)$），即求出 $\delta(S)$。

代码实现细节

对于块长为 b 的分块，相当于是建立一个 $(x,y) \to \left(\left\lfloor \frac{x}{b} \right\rfloor, \left\lfloor \frac{y}{b} \right\rfloor\right)$ 的映射。

代码实现细节

对于块长为 b 的分块，相当于是建立一个 $(x, y) \to \left(\left\lfloor \frac{x}{b} \right\rfloor, \left\lfloor \frac{y}{b} \right\rfloor\right)$ 的映射。

对 $\left(\left\lfloor \frac{x}{b} \right\rfloor, \left\lfloor \frac{y}{b} \right\rfloor\right)$ 建哈希表，将每个点存到对应的块中，哈希表的查询复杂度可以认为是期望 $\mathcal{O}(1)$。

代码实现细节

对于块长为 b 的分块，相当于是建立一个 $(x,y)\to\left(\left\lfloor\frac{x}{b}\right\rfloor,\left\lfloor\frac{y}{b}\right\rfloor\right)$ 的映射。

对 $\left(\left\lfloor\frac{x}{b}\right\rfloor,\left\lfloor\frac{y}{b}\right\rfloor\right)$ 建哈希表，将每个点存到对应的块中，哈希表的查询复杂度可以认为是期望 $\mathcal{O}(1)$。

查询 *8*-邻居 块和自身块相当于查询块 $\left(\left\lfloor\frac{x}{b}\right\rfloor+\Delta x,\left\lfloor\frac{y}{b}\right\rfloor+\Delta y\right)$，其中 $\Delta x,\Delta y\in\{-1,0,1\}$。

代码实现细节

对于块长为 b 的分块，相当于是建立一个 $(x,y) \to \left(\left\lfloor \frac{x}{b} \right\rfloor, \left\lfloor \frac{y}{b} \right\rfloor\right)$ 的映射。

对 $\left(\left\lfloor \frac{x}{b} \right\rfloor, \left\lfloor \frac{y}{b} \right\rfloor\right)$ 建哈希表，将每个点存到对应的块中，哈希表的查询复杂度可以认为是期望 $\mathcal{O}(1)$。

查询 *8*-邻居 块和自身块相当于查询块 $\left(\left\lfloor \frac{x}{b} \right\rfloor + \Delta x, \left\lfloor \frac{y}{b} \right\rfloor + \Delta y\right)$，其中 $\Delta x, \Delta y \in \{-1, 0, 1\}$。

需要处理好重点和哈希冲突的情况。

时间复杂度

先分析筛选过程的时间复杂度，检查邻居只需要检查 $\mathcal{O}(1)$ 个点，因此每轮迭代的时间复杂度是 $\mathcal{O}(|S_i|)$。特别地，我们有：

时间复杂度

先分析筛选过程的时间复杂度，检查邻居只需要检查 $\mathcal{O}(1)$ 个点，因此每轮迭代的时间复杂度是 $\mathcal{O}(|S_i|)$。特别地，我们有：

$$\mathbb{E}\left(\sum_{i=1}^{i^*}|S_i|\right) \leq 2n \tag{1}$$

时间复杂度

先分析筛选过程的时间复杂度，检查邻居只需要检查 $\mathcal{O}(1)$ 个点，因此每轮迭代的时间复杂度是 $\mathcal{O}(|S_i|)$。特别地，我们有：

$$\mathbb{E}\left(\sum_{i=1}^{i^*}|S_i|\right) \leq 2n \tag{1}$$

证明：给出一个粗略的估计，将 S_i 中的点按照 $d(x)$ 的大小升序排序，其中选取到第 j 名至多能在 S_{i+1} 中保留 $j-1$ 个点，因此

时间复杂度

先分析筛选过程的时间复杂度，检查邻居只需要检查 $\mathcal{O}(1)$ 个点，因此每轮迭代的时间复杂度是 $\mathcal{O}(|S_i|)$。特别地，我们有：

$$\mathbb{E}\left(\sum_{i=1}^{i^*}|S_i|\right) \leq 2n \tag{1}$$

证明：给出一个粗略的估计，将 S_i 中的点按照 $d(x)$ 的大小升序排序，其中选取到第 j 名至多能在 S_{i+1} 中保留 $j-1$ 个点，因此

$$\mathbb{E}\left(|S_{i+1}|\right) \leq \frac{1}{|S_i|}\sum_{j=0}^{|S_i|-1} j = \frac{|S_i|-1}{2} \tag{2}$$

时间复杂度

先分析筛选过程的时间复杂度，检查邻居只需要检查 $\mathcal{O}(1)$ 个点，因此每轮迭代的时间复杂度是 $\mathcal{O}(|S_i|)$。特别地，我们有：

$$\mathbb{E}\left(\sum_{i=1}^{i^*}|S_i|\right) \le 2n \tag{1}$$

证明：给出一个粗略的估计，将 S_i 中的点按照 $d(x)$ 的大小升序排序，其中选取到第 j 名至多能在 S_{i+1} 中保留 $j-1$ 个点，因此

$$\mathbb{E}\left(|S_{i+1}|\right) \le \frac{1}{|S_i|}\sum_{j=0}^{|S_i|-1} j = \frac{|S_i|-1}{2} \tag{2}$$

每次筛选是O｜Si｜
的用hash检查邻域

也就是说每轮迭代 S_i 期望减少一半。

时间复杂度

将 (2) 式带回 (1) 式即证

时间复杂度

将 (2) 式带回 (1) 式即证

$$\mathbb{E}\left(\sum_{i=1}^{i^*}|S_i|\right)=\sum_{i=1}^{i^*}\mathbb{E}\left(|S_i|\right)\leq\sum_{i=1}^{n}\frac{n}{2^{i-1}}\leq 2n$$

$\square$

时间复杂度

将 (2) 式带回 (1) 式即证

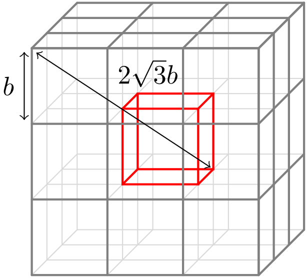
$$\mathbb{E}\left(\sum_{i=1}^{i^*}|S_i|\right)=\sum_{i=1}^{i^*}\mathbb{E}\left(|S_i|\right)\leq\sum_{i=1}^{n}\frac{n}{2^{i-1}}\leq 2n$$

□

对于第二部分求解 $\delta(S)$ 的过程，每个点只会检查 $\mathcal{O}(1)$ 个相邻点，所以这部分的时间复杂度也是线性的。总时间复杂度 $\mathcal{O}(n)$。

如何推广到三维

推广到三维是平凡的，我们照猫画虎地把检查 *8-邻居* 改成检查 *26-邻居* 就好了。但是三维中用 $b = d(x_i)/3$ 会出问题，因为立方体中能塞下的最长距离是 $2\sqrt{3}/3 > 1$，无法保证 $d(x_i)$ 递减。



只需要把块长调整为 $b = d(x_i)/4$，此时 $\sqrt{3}/2 < 1$ 满足条件。

References

[1] S. Khuller and Y. Matias. "A Simple Randomized Sieve Algorithm for the Closest-Pair Problem". In: Information and Computation 118.1 (1995), pp. 34–37. ISSN: 0890-5401. DOI: https://doi.org/10.1006/inco.1995.1049. URL: https://www.sciencedirect.com/science/article/pii/S0890540185710498.

THANK YOU!