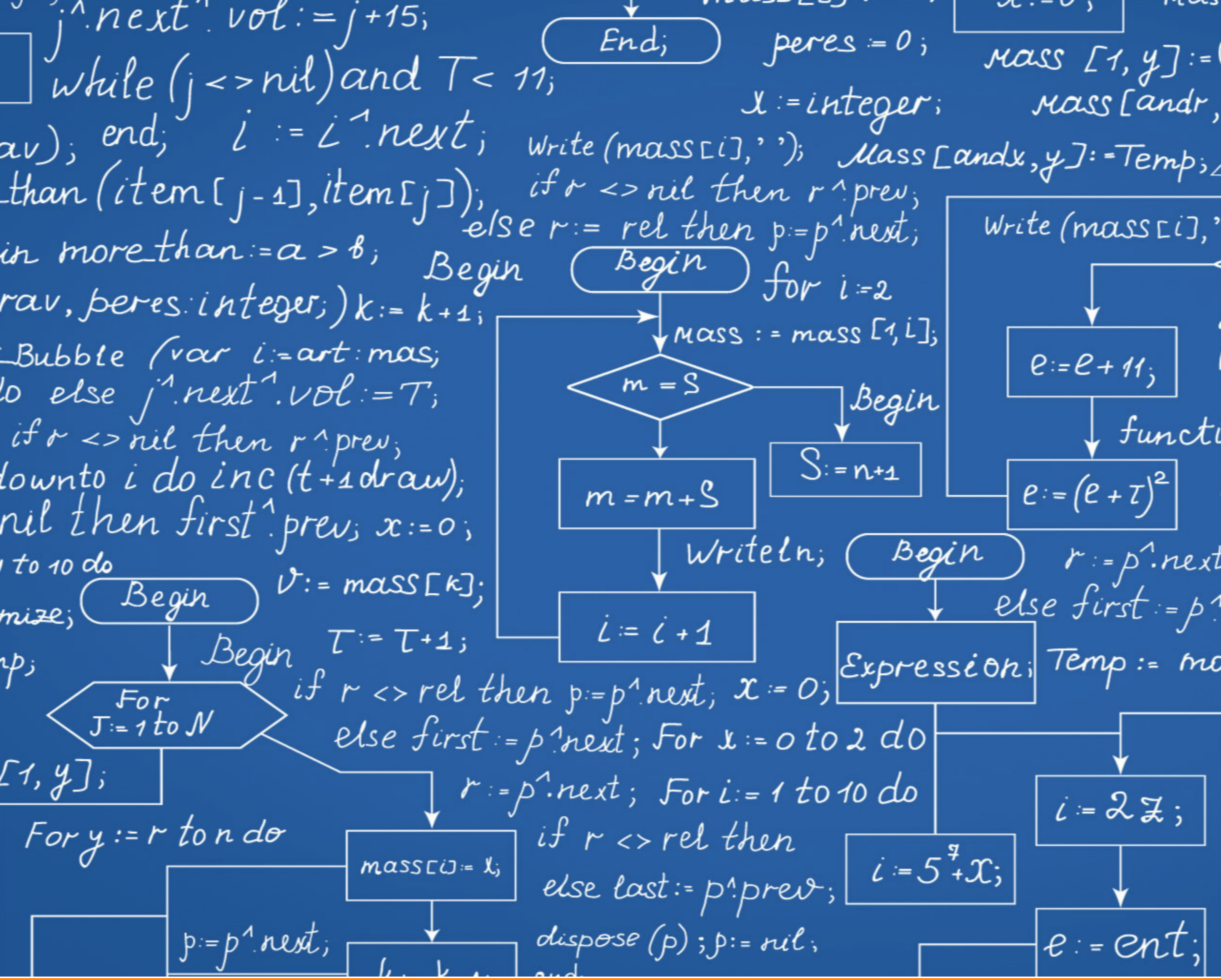

何老师算法课笔记

还没有想好的副标题

作者：计卓 1801 全体

组织：华中科技大学

时间：2020 年 11 月 10 日

版本：1.0.0

目录

1 分治算法之平面最近点对问题 **B**

1.1 平面最近点对问题定义 . . . . B

1.2 分治算法设计及伪代码 . . . . B

1.3 分治算法正确性证明与复杂度分析 . . . . D

1.4 参考代码（C++ 实现） . . . . D

第 1 章 分治算法之平面最近点对问题

内容提要

- 平面最近点对问题定义
- 分治算法正确性证明及复杂度分析
- 分治算法设计及伪代码
- 参考代码（C++）

1.1 平面最近点对问题定义

给定二维平面上的 $n(n \geq 2)$ 个不同的点 p 组成点集 $P = \{p_i | 1 \leq i \leq n\}$，设计算法寻找欧式距离最近的点对 (A, B)。

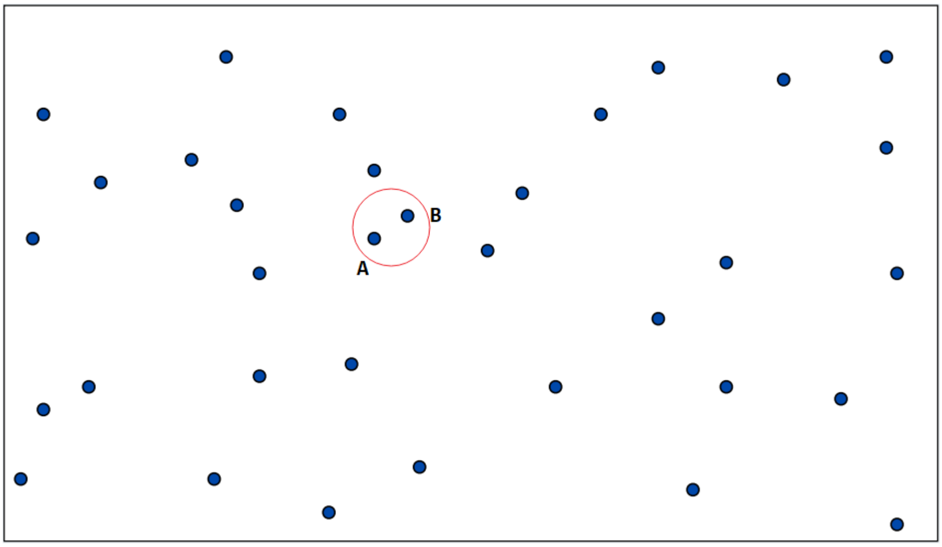

图 1.1: 问题定义图例

如上图Figure 1.1中点对 (A, B) 即为问题的答案。

1.2 分治算法设计及伪代码

对于这样一个问题，我们很直接地可以使用 BF (Brute Force) 算法进行暴力求解，即二重循环计算所有点之间的距离，从而获得最小距离，显然该算法的时间复杂度为 $O(n^2)$。那么有没有更快的算法呢？本章我们使用经典的算法思想——分治，设计一个 $O(n \log n)$ 的算法。

1.2.1 分治问题

遵循分治思想，我们首先要考虑如何分治问题使得问题规模约减。

我们使用 X 坐标作为第一关键字、Y 坐标作为第二关键字，对点集 P 进行排序，并以点 $p_{\lfloor \frac{n}{2} \rfloor}$ 作为分治点，获得如下两个点集：

$$P_1 = \{p_i \mid 1 \leq i \leq \lfloor \frac{n}{2} \rfloor\}$$

$$P_2 = \{p_i \mid \lfloor \frac{n}{2} \rfloor < i \leq n\}$$

这样就将当前问题约减为两个规模为 $\lfloor \frac{n}{2} \rfloor$ 的子问题。分治过程如Figure 1.2中所示。

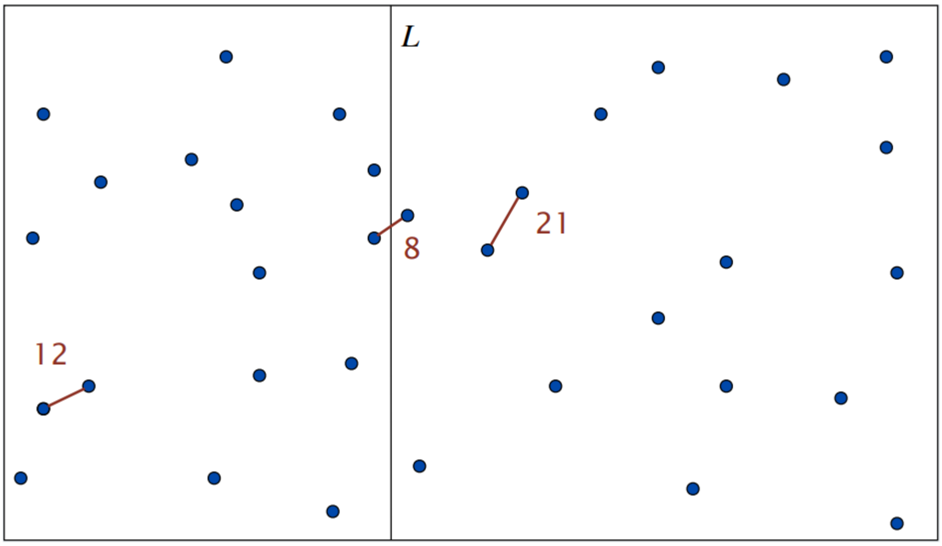

图 1.2: 分治过程图例

如此递归下去，我们可以求得两个点集相对应的最近点对距离 δ_1, δ_2，取其中较小值记为 $\delta = \min\{\delta_1, \delta_2\}$。

1.2.2 合并结果

接着，我们需要考虑如何合并子问题的解。

上述的 δ 一定是正确的合并结果嘛？显然不是，我们并没有考虑，一端在 P_1，一端在 P_2 的线段。因此，在合并阶段，我们要将这种情况考虑在内。

这里，我们将所有横坐标与分治点 $p_{\lfloor \frac{n}{2} \rfloor}$ 的横坐标 $x_{\lfloor \frac{n}{2} \rfloor}$ 差值小于 δ 的点组成集合 B，即

$$B = \{p_i \mid \left| x_i - x_{\lfloor \frac{n}{2} \rfloor} \right| \leq \delta,\ 1 \leq i \leq n\}$$

因为只有 B 集合中的点之间的距离才有可能小于 δ。B 集合如下图Figure 1.3中阴影部分所示:

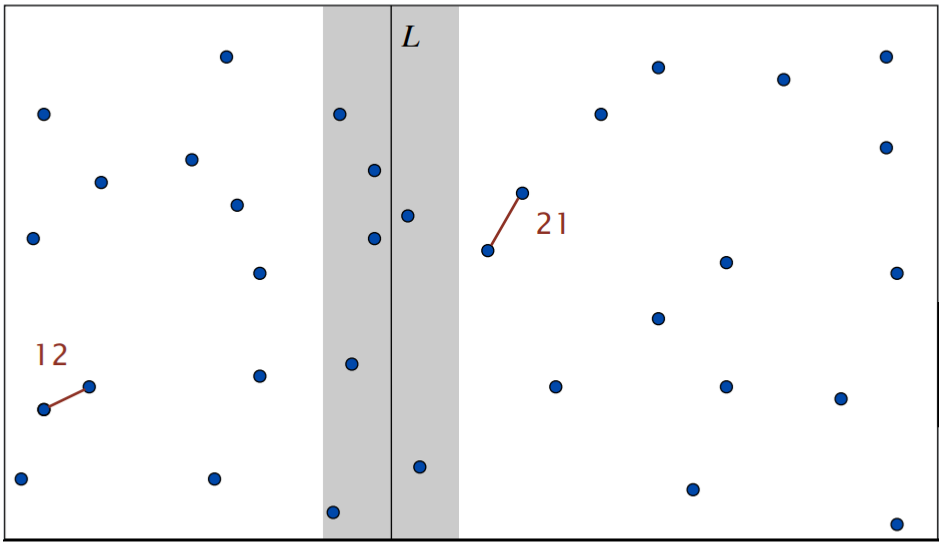

图 1.3: 合并过程图例

进一步，我们的目标是检验在 B 集合中是否存在距离比 δ 更近的点对，以此更新当前问题的解。因此，对于每个 $p_i = (x_i, y_i) \in B$ 遍历所有在其之下竖直距离不超过 δ 的点，即遍历集合

$$C(p_i) = \{p_j \mid y_i - \delta \leq y_j \leq y_i, p_j \in B\}$$

为了方便遍历，这里需要对 B 集合中的点，以 Y 坐标为第一关键字，X 坐标为第二关键字，进行排序。

至此，我们完成了父问题的分治与子问题的合并。

1.2.3 伪代码

Algorithm 1: Nearest-Pair

```
Data: Point Set P = {p_i | 1 ≤ i ≤ n, p_i = (x_i, y_i)}
Result: the nearest pair (A, B)
begin
    Sort points in P by x-coordinate in descending order
    m ← ⌊n/2⌋
    δ_1 ← Nearest-Pair(P[1, ..., m])
    δ_2 ← Nearest-Pair(P[m+1, ..., n])
    δ ← min{δ_1, δ_2}
    B ← {}
    foreach p_i ∈ P do
        if |x_i − x_m| ≤ δ then
            add p_i to B
    Sort points in B by y-coordinate in descending order
    for i ← 1 to |B| do
        for j ← i+1 to |B| do
            if |y_i − y_j| ≤ δ then
                δ ← min{δ, Euclidean-Distance(p_i, p_j)}
    Return δ
```

1.3 分治算法正确性证明与复杂度分析

1.4 参考代码（C++ 实现）

参考文献