07-D2

搜索树应用

多层搜索树：二维及多维

几株不知名的树，已脱下了黄叶
只有那两三片，多么可怜在枝上抖怯
它们感到秋来到，要与世间离别

邓俊辉
deng@tsinghua.edu.cn

二维范围查询 = x-查询 + y-查询

❖ 推而广之，每一次m维的正交范围查询

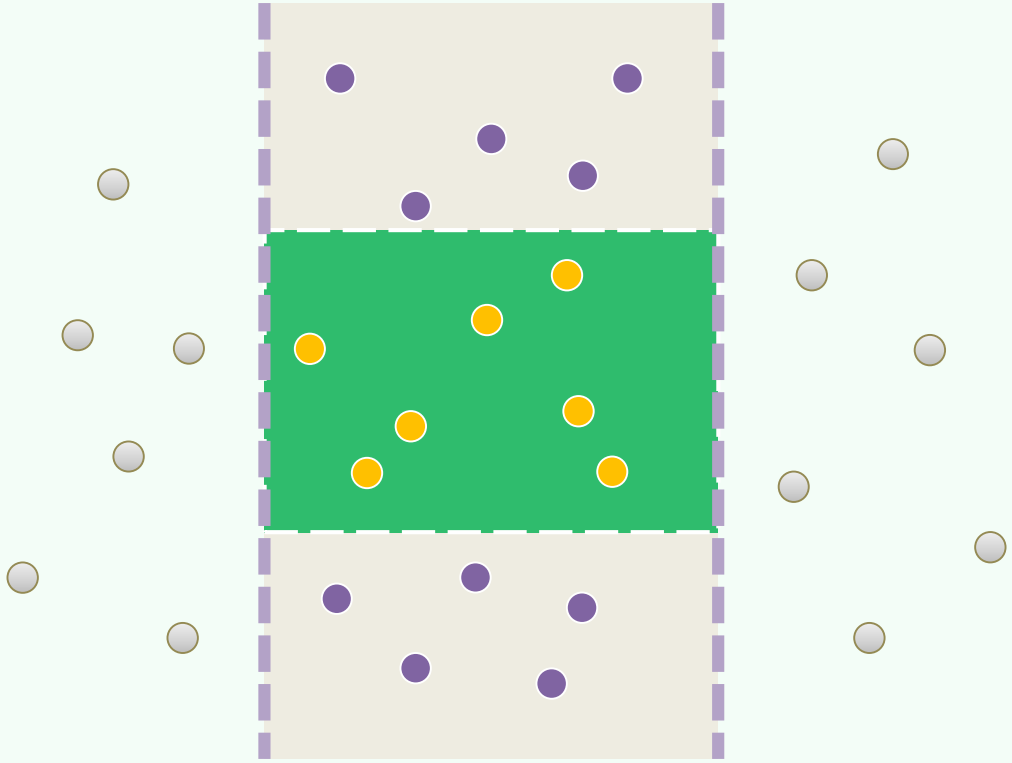
都可通过m次的1维正交查询来回答

❖ 比如，每一次2维范围查询

都可由2次的1维正交查询来回答

- 先查找出沿x轴落在$[x_1, x_2]$内的点，进而
- 再筛选出沿y轴落在$[y_1, y_2]$内的点

❖ 事情果然就如此简单?



最坏情况

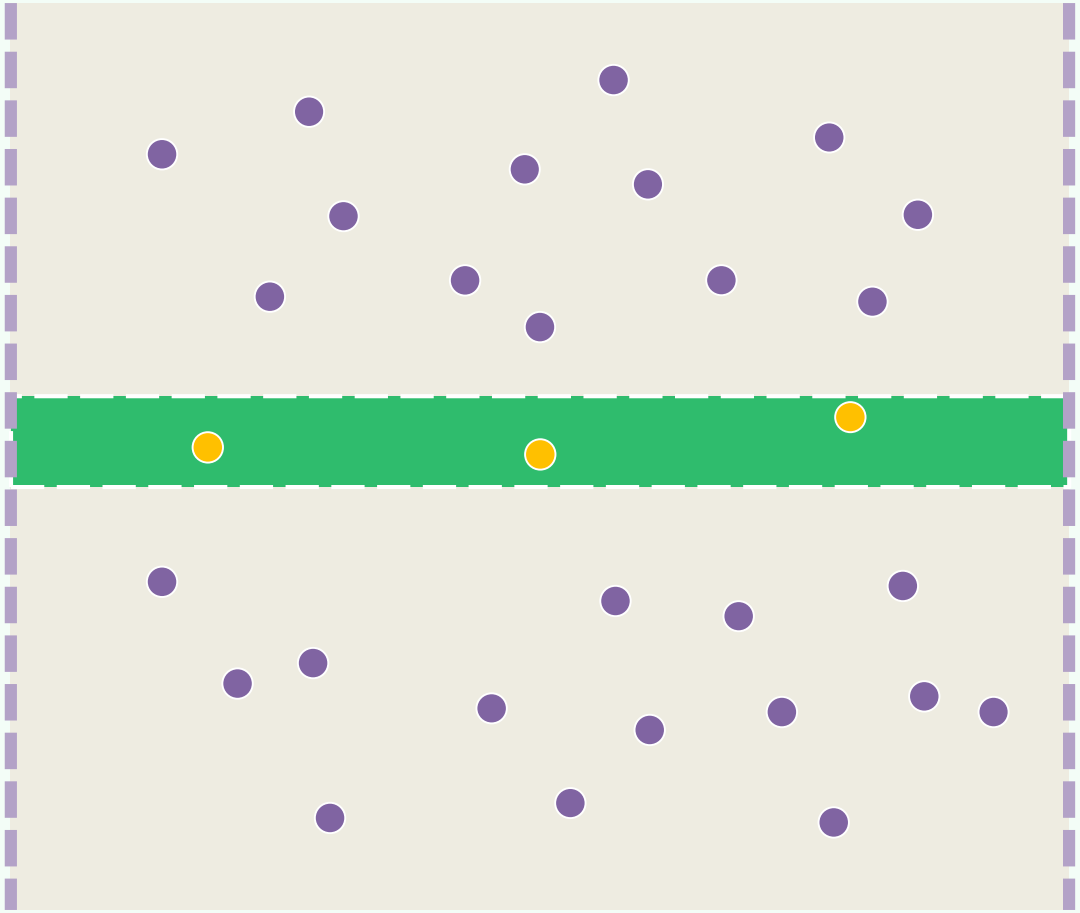
❖ 完全有可能...

- x查询虽命中了（几乎）所有点，而接下来
- y查询却排除了（几乎）所有点

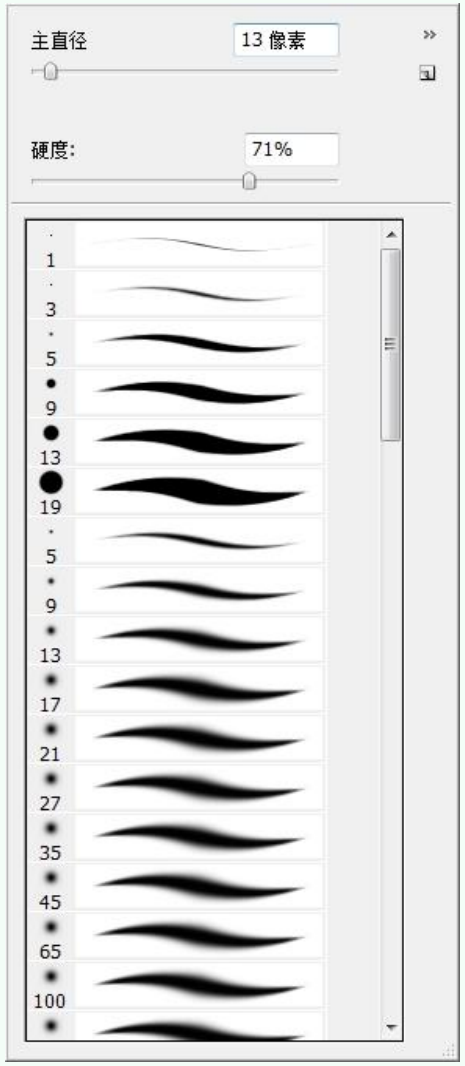
❖ 如此，花费Ω(n)时间却几乎一无所获

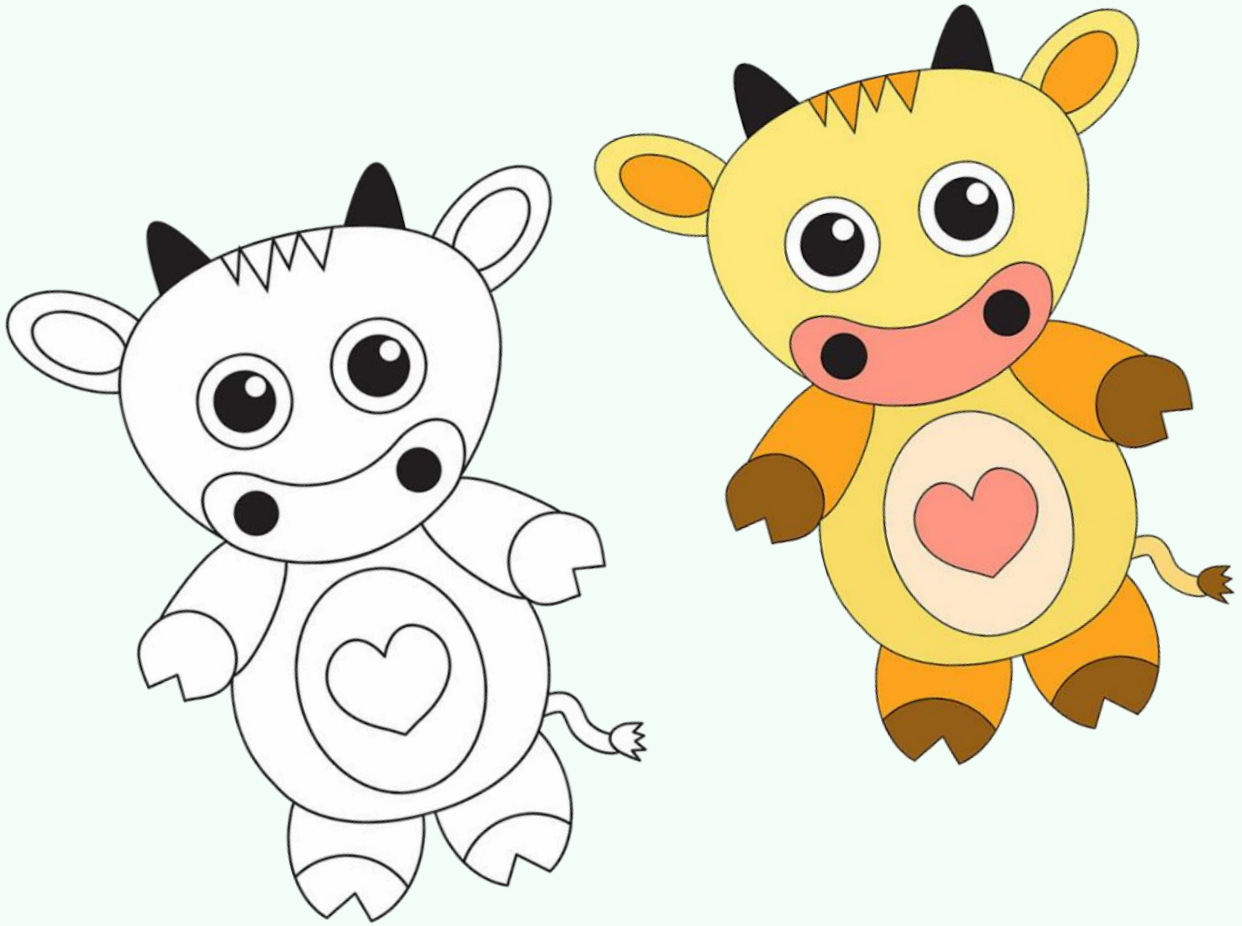
输出量（几乎）为0

❖ 如何做到输出敏感（output sensitivity）？



艺术与技术


主直径
13 像素
硬度:
71%
1
3
5
9
13
19
5
9
13
17
21
27
35
45
65
100

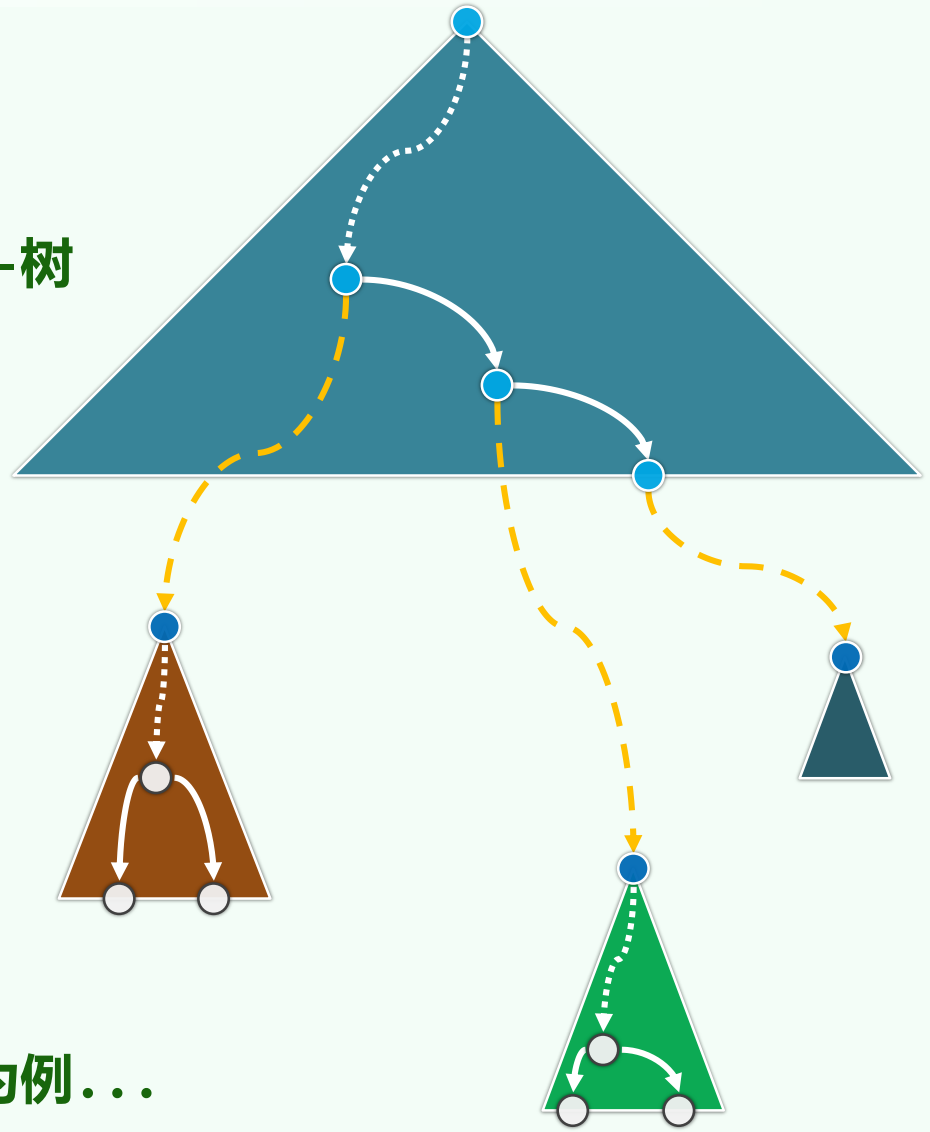



MICRON
MICRON
MICRON

MLST = x-Tree * y-Trees

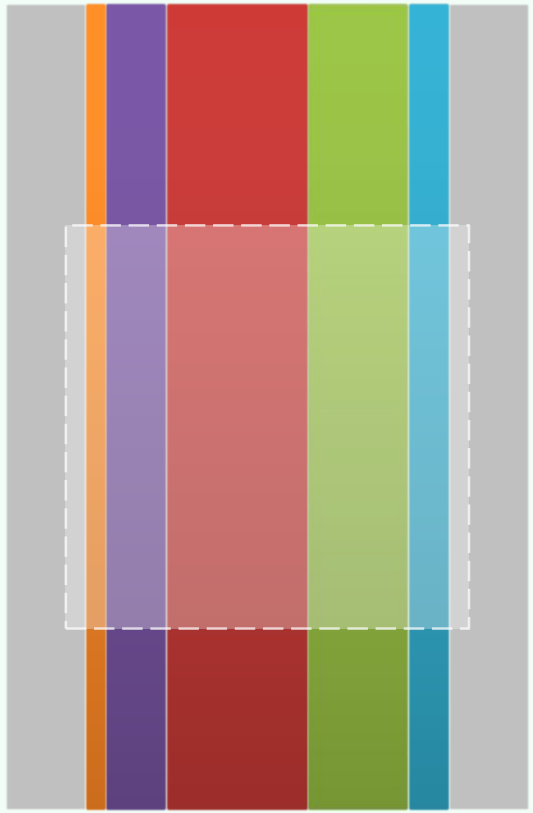
- 首先，针对x-查询，构建一棵1维BBST（x-树）
- 然后，对于x-树中的任何一棵子树v，另外构造一棵关联的y-树
  - 二者对应于输入点集的同一子集，只不过
  - 顾名思义，y-树是将这些点按y-方向排序
- 每棵x-子树，可以通过引用，直接找到与之对应的y-树
- 如果还有更多维度，也可照此逐层推广

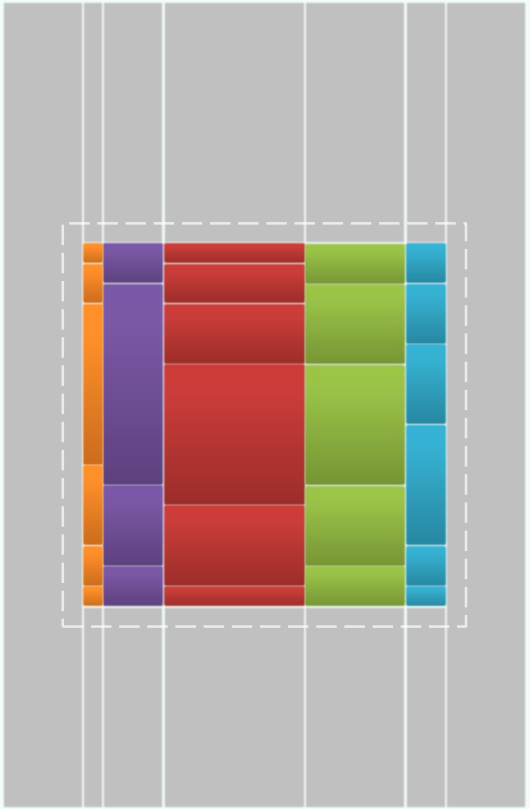
  整个结构也称作多层搜索树（Multi-Level Search Tree）
- 那么，如何借助MLST来高效地完成范围查询呢？以下以2维为例...

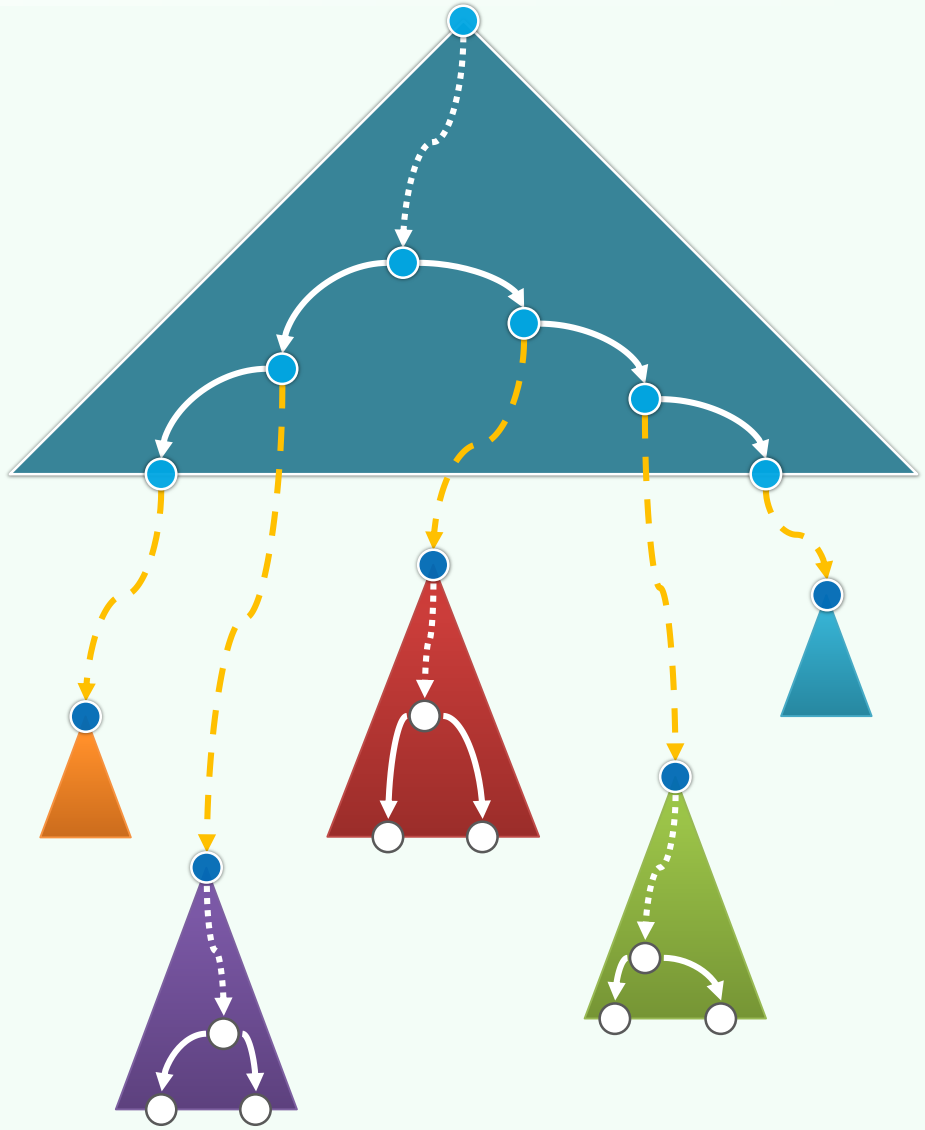


二维范围查询 = x-查询 * y-查询

❖ 查询时间 = $\mathcal{O}(r + \log^2 n)$ ~ $\mathcal{O}(r + \log n)$







查询算法

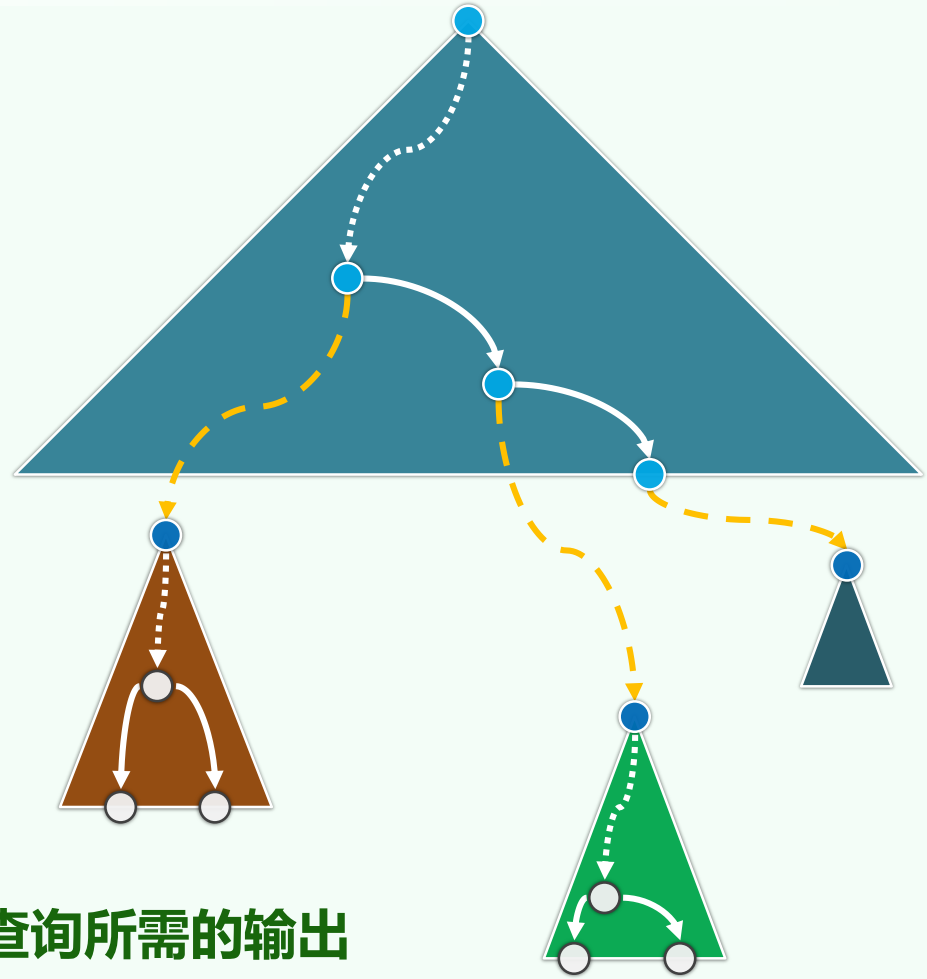
❖ 过程

- 首先，通过x-query(x_1,x_2)

  从x-树中找出$O(\log n)$棵x-子树

- 再分别在与之对应的$O(\log n)$棵y-树中

  通过y-query(y_1,y_2)，分批地筛选出所有命中的点



❖ 性质

- x-query和y-query都是基本的一维范围查询
- y-query各自输出的子集相互无交，且其并集正是原查询所需的输出
- 除却输出本身所需的$O(r)$时间，x-query及所有的y-query总共只需$O(\log n)$时间

复杂度：查询时间

❖ 声明：

在将平面上的任何一组点整理为一棵2层的MLST之后

每次2维范围查询都只需要 $\mathcal{O}(r + \log^2 n)$ 时间

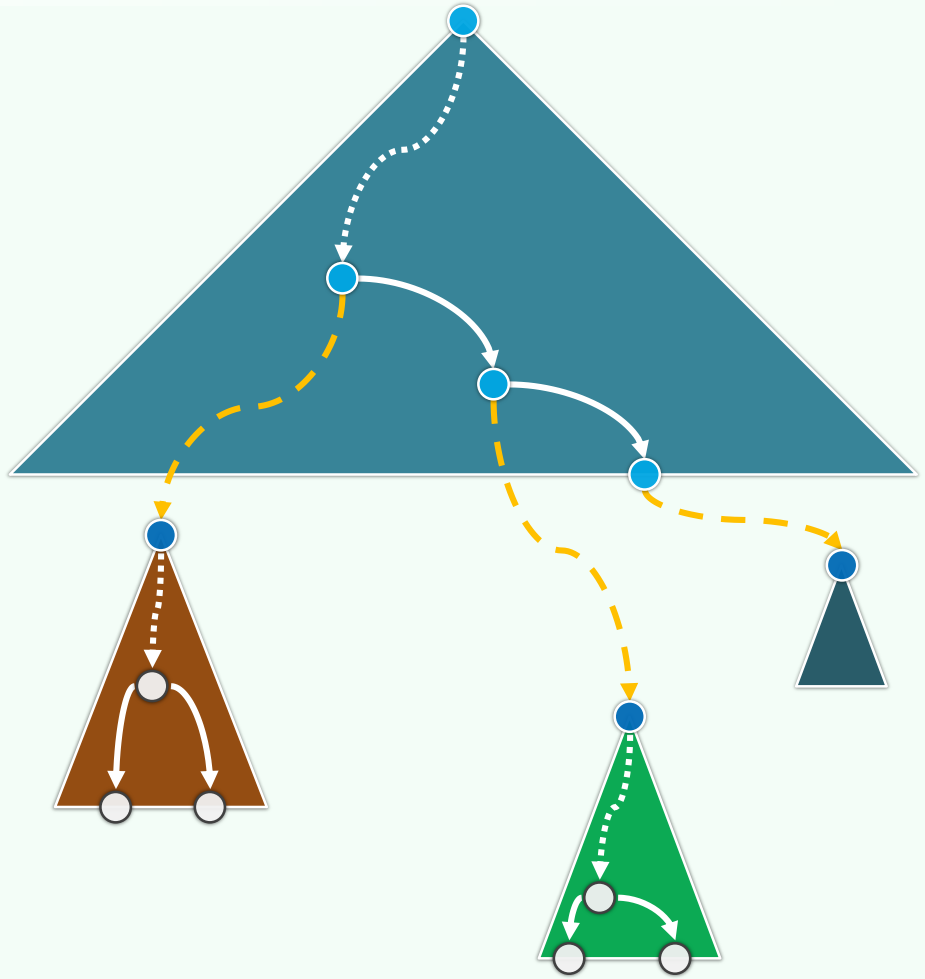
❖ 证明：

- x-query可在 $\mathcal{O}(\log n)$ 时间内

  锁定 $\mathcal{O}(\log n)$ 棵x-子树

- 接下来的 $\mathcal{O}(\log n)$ 次y-query

  各自也仅需 $\mathcal{O}(\log n)$ 时间



复杂度：预处理 + 存储空间

- 对于平面上任意的n个点

  A> 都可以在 $\mathcal{O}(n \log n)$ 时间内建立一棵2层的MLST

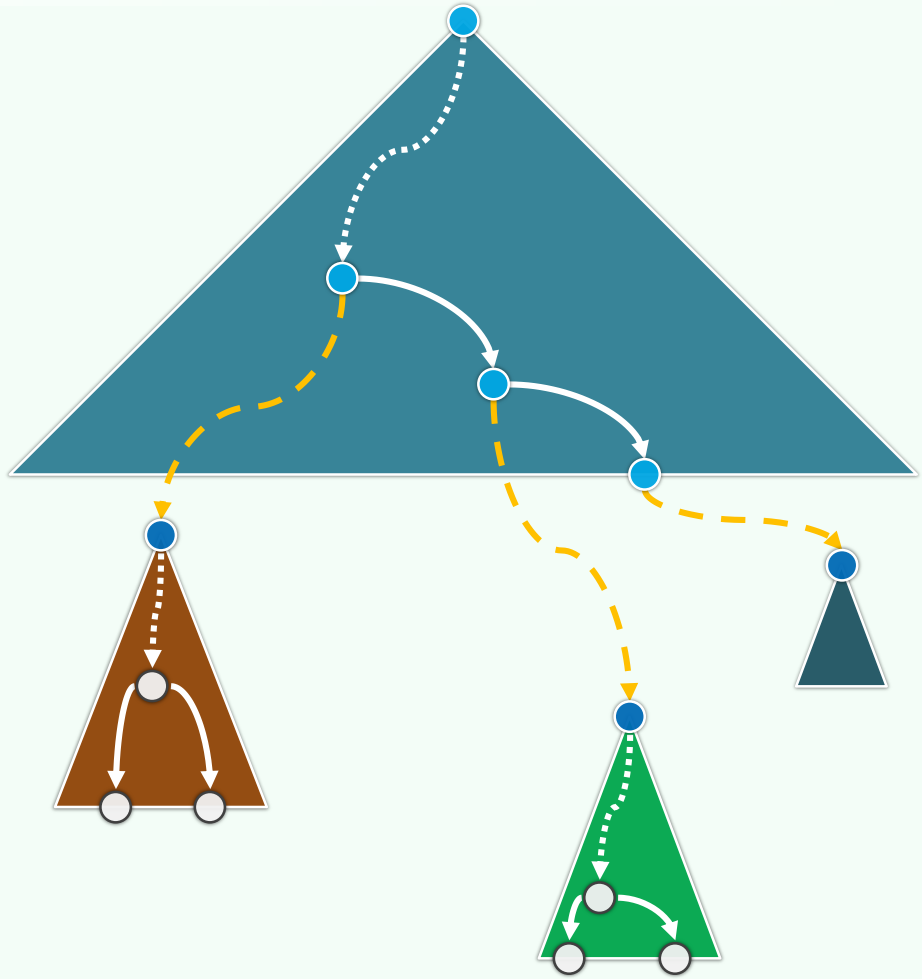
  B> 该结构只需 $\mathcal{O}(n \log n)$ 空间

- 为做到A，可以“**自底而上、逐层合并**”的方式来构造

- 为理解B，只需注意到

  输入的每一个点，都记录在 $\mathcal{O}(\log n)$ 棵y-树中

- 也可以按照**深度**，对y-树**分类**统计...



更高维度

- 由欧氏空间$\mathcal{E}^d$（d≥2）中的任意n个点，都可以

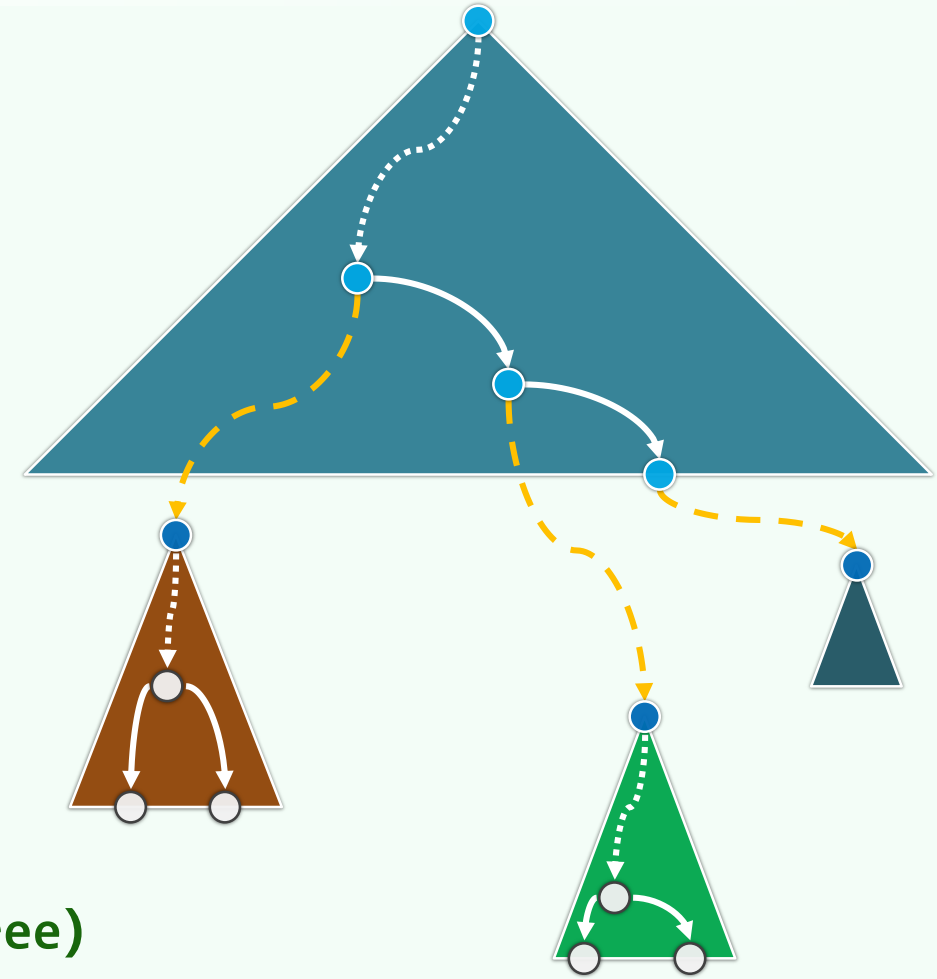
  A> 在 $\mathcal{O}(n \cdot \log^{d-1} n)$ 时间内

  构造出一棵d层的MLST

  B> 该结构只需 $\mathcal{O}(n \cdot \log^{d-1} n)$ 空间

  C> 借助该结构，每次d维范围查询

  都可以在 $\mathcal{O}(r + \log^{d} n)$ 时间内完成



- 借助分散层叠的技巧，将MLST升级为范围树（Range Tree）

  即可将C>改进至 $\mathcal{O}(r + \log^{d-1} n)$ ...