07-B6

BST Application

kd-Tree: Complexity

肉眼看不清细节，但他们都知道那是木星所在的位置，这颗太阳系最大的行星已经坠落到二维平面上了。

有人嘲笑这种体系说：为了能发现这个比例中项并组成政府共同体，按照我的办法，只消求出人口数字的平方根就行了。

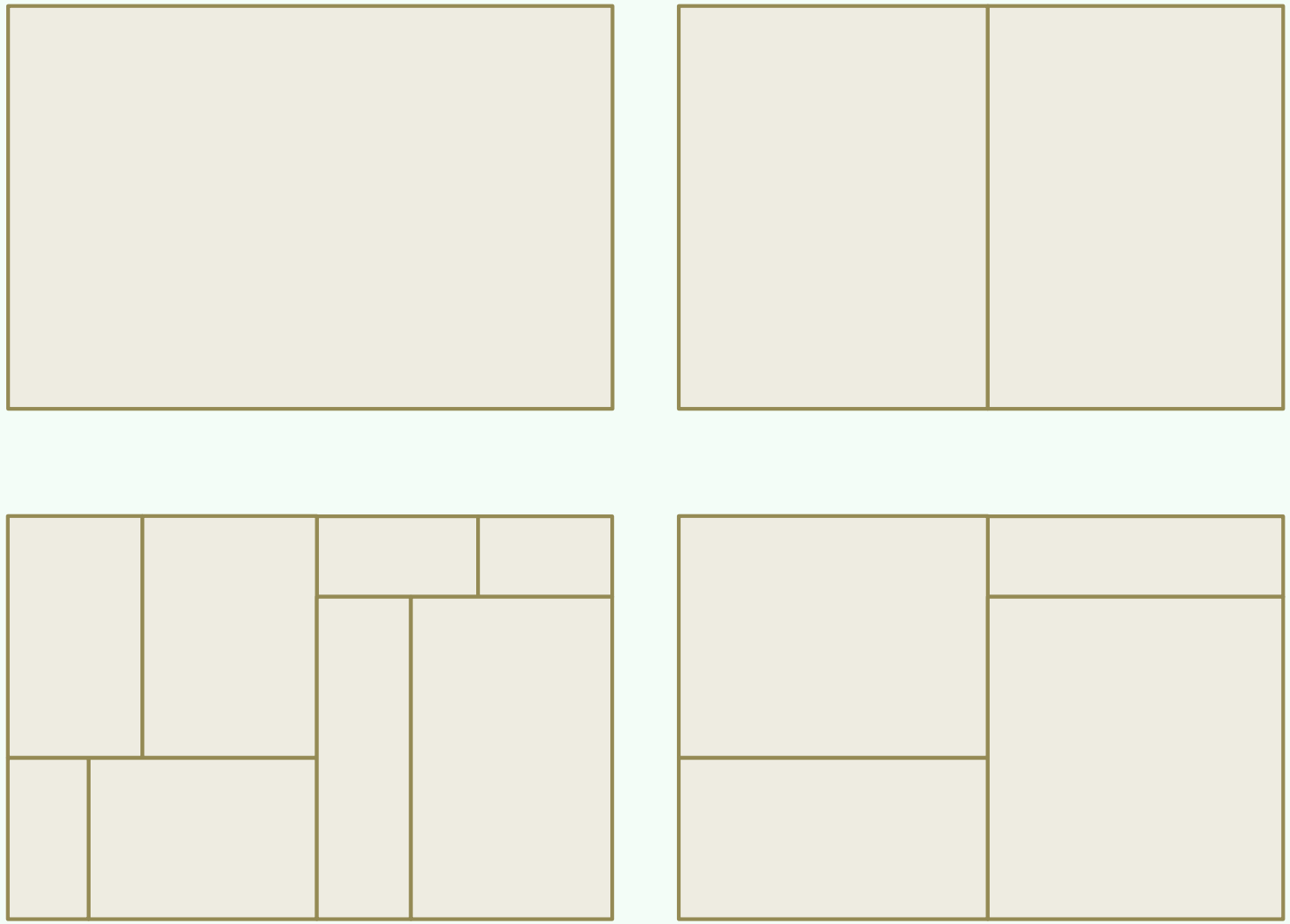
邓俊辉
deng@tsinghua.edu.cn

Preprocessing

- T(n)

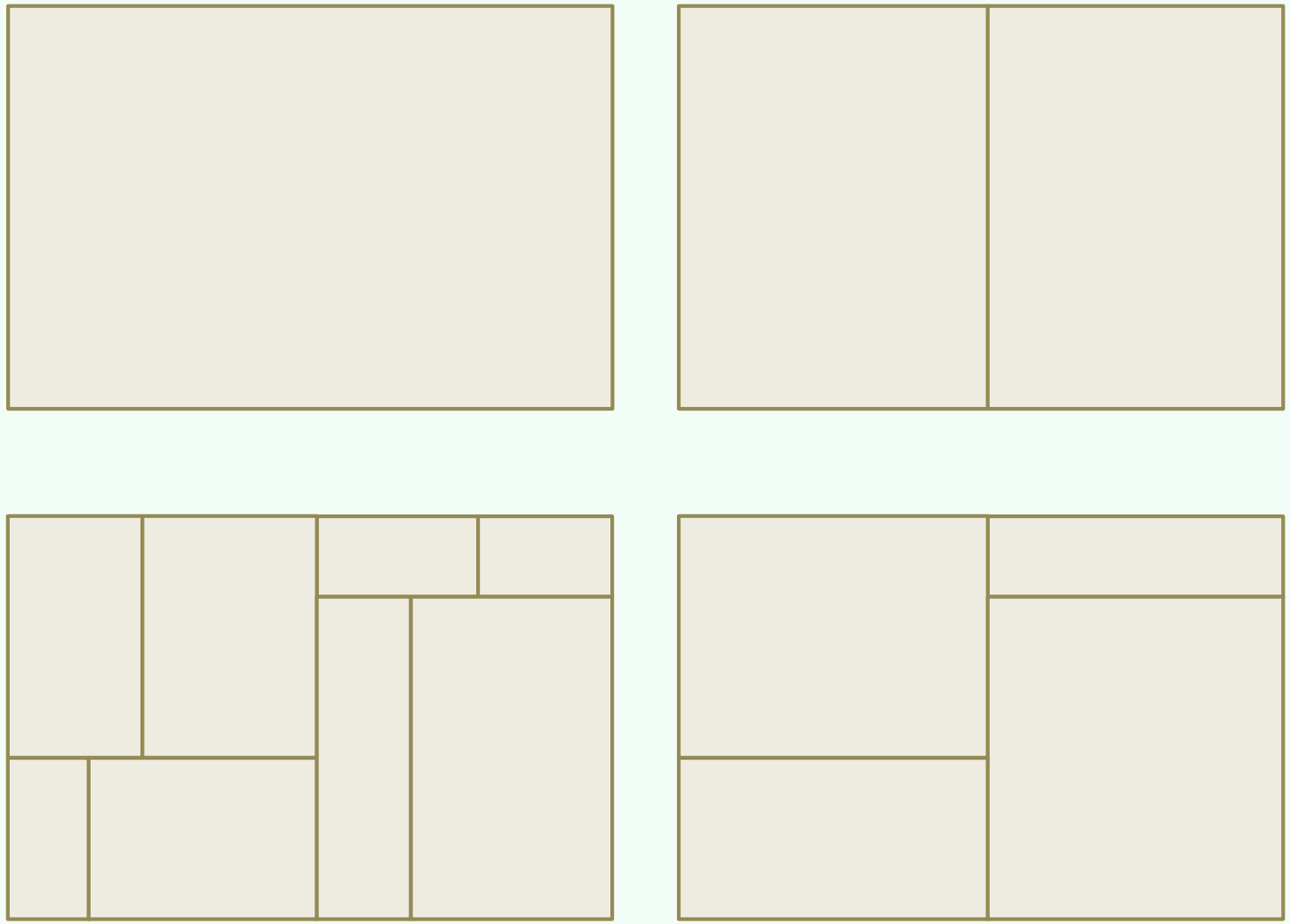
  = 2*T(n/2) + $\mathcal{O}$(n)

  = $\mathcal{O}$(nlogn)



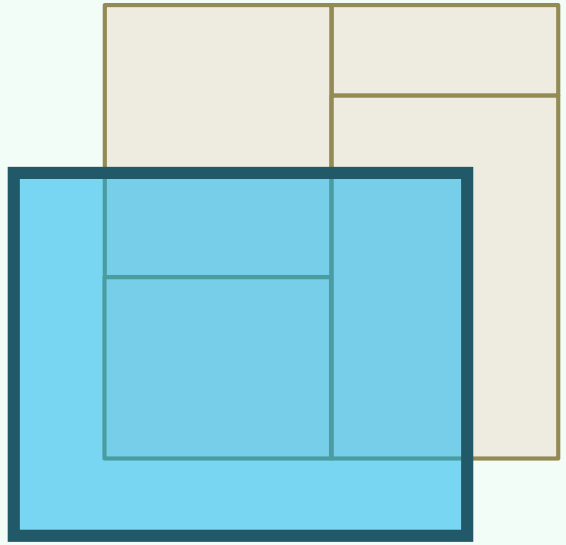
Storage

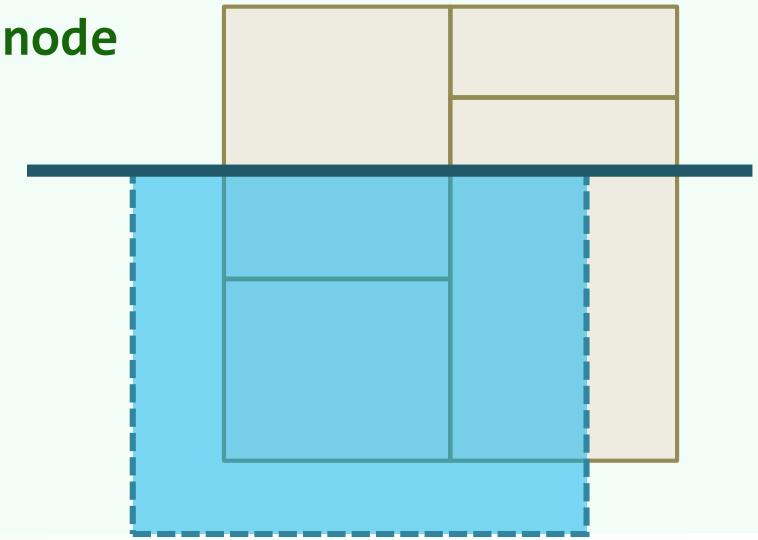
- The tree has a height of $O(\log n)$
- $1 + 2 + 4 + \ldots + O(2^{\log n}) = O(n)$

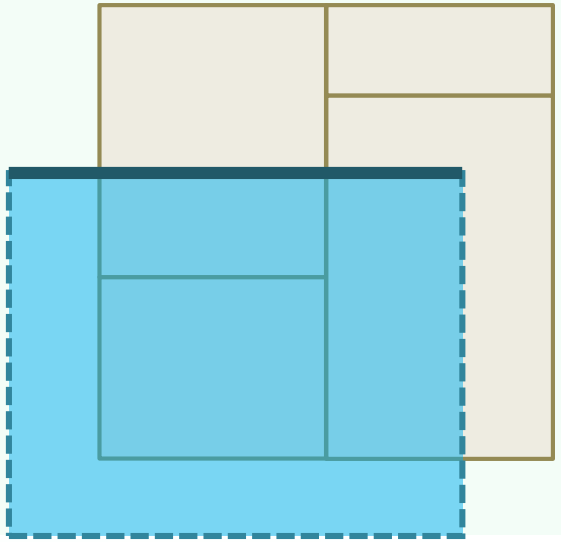


Query Time

- Claim: Report + Search = $\mathcal{O}(r + \sqrt{n})$
- The searching time depends on Q(n), the number of
  - recursive calls, or
  - sub-regions intersecting with R (at all levels)
- No more than 2 of the 4 grandchildren of each node will recurse
  - $Q(1) = \mathcal{O}(1)$
  - $Q(n) = 2 \cdot Q(n/4) + \mathcal{O}(1)$
- Solve to $Q(n) = \mathcal{O}(\sqrt{n})$







Beyond 2D

- Can 2d-tree be extended to kd-tree and help HIGHER dimensional GRS?

  If yes, how efficiently can it help?

- A kd-tree in k-dimensional space is constructed by

  recursively divide $\mathcal{E}^d$ along the 1st, 2nd, ..., kth dimensions

- An orthogonal range query on a set of n points in $\mathcal{E}^d$
  - can be answered in $\mathcal{O}(r + n^{1-1/d})$ time,
  - using a kd-tree of size $\mathcal{O}(n)$, which
  - can be constructed in $\mathcal{O}(n \log n)$ time