2D Range Query

KD-tree and Range tree
2D Orthogonal Range Query

- Data: A set $S$ of $n$ points
- Query: Report/Count subset of $S$ that lie in a rectangle range
K-d Tree
K-d Tree
K-d Tree
K-d Tree
K-d Tree
K-d Tree
K-d Tree
K-d Tree
K-d Tree
K-d Tree
K-d Tree
K-d Tree

- Space: $O(n)$
K-d Tree

• If t is a node:
  – t.val: cut value
  – t.dir: cut direction
  – t.left, t.right: child

• If t is a leaf:
  – t.pt: point
K-d Tree

• If t is a node:
  – t.val: cut value
  – t.dir: cut direction
  – t.left, t.right: child
• If t is a leaf:
  – t.pt: point

BuildTree (S, d)  //d: direction
1. If |S|=1, return leaf t where
   1. t.pt is the point of S
   2. x be median of d-th coordinates of all points in S
   3. L (R) be subset of S whose d-th coordinates are no greater than (greater than) x
   4. Return node t where
      1. t.val = x
      2. t.dir = d
      3. t.left = BuildTree (L, 3-d)
      4. t.right = BuildTree (R, 3-d)

T(n) = O(n) + 2T(n/2)
= O(n log n)
K-d Tree

- Space: $O(n)$
- Build time: $O(n \log n)$
Query a K-d tree
Range of a node
Range of a node
Range of a node
Range of a node
Query a K-d tree
Query a K-d tree
Query a K-d tree
Query a K-d tree
Query a K-d tree
Query a K-d tree
Query a K-d tree

Range of * lies within the query range.
The whole subtree rooted at * is reported.
Query a K-d tree

- If t is a node:
  - t.val: cut value
  - t.dir: cut direction
  - t.left, t.right: child
  - t.range: range

- If t is a leaf:
  - t.pt: point

\[\text{Query} \ (t, r) \quad \text{//} r: \text{query range}\]

1. If t is a leaf
   1. If t.pt is inside r, return t.pt
   2. Else return NULL
2. If t.range is inside r
   1. ReportTree (t)
3. Else if t.range intersects r
   1. Return \text{Query} \ (t.left, r)
   2. Query (t.right, r)
Query a K-d tree

• Complexity:
  – Total time for (1-2): O(k)
  – # calls to Query: ???

```plaintext
Query (t, r)    //r: query range
1. If t is a leaf
   1. If t.pt is inside r, return t.pt
   2. Else return NULL
2. If t.range is inside r
   1. ReportTree (t)
3. Else if t.range intersects r
   1. Return Query (t.left, r)
   Query (t.right, r)
```
Query a K-d tree

- Query(t, r) is called if t’s parent’s range intersects (but does not lie inside) r

Such range must contain a border edge of r
Query a K-d tree

• How many nodes of a k-d tree can be “stabbed” by a line (i.e., the line passes through the node’s range)?
Query a K-d tree

• How many nodes of a k-d tree can be “stabbed” by a line (i.e., the line passes through the node’s range)?

<table>
<thead>
<tr>
<th>Tree level</th>
<th># stabbed nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Query a K-d tree

• How many nodes of a k-d tree can be “stabbed” by a line (i.e., the line passes through the node’s range)?

<table>
<thead>
<tr>
<th>Tree level</th>
<th># stabbed nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>
Query a K-d tree

• How many nodes of a k-d tree can be “stabbed” by a line (i.e., the line passes through the node’s range)?

<table>
<thead>
<tr>
<th>Tree level</th>
<th># stabbed nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>
Query a K-d tree

- How many nodes of a k-d tree can be “stabbed” by a line (i.e., the line passes through the node’s range)?

<table>
<thead>
<tr>
<th>Tree level</th>
<th># stabbed nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>
Query a K-d tree

• How many nodes of a k-d tree can be “stabbed” by a line (i.e., the line passes through the node’s range)?

<table>
<thead>
<tr>
<th>Tree level</th>
<th># stabbed nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>$k$</td>
<td>$2^{\lfloor k/2 \rfloor}$</td>
</tr>
</tbody>
</table>
• How many nodes of a k-d tree can be “stabbed” by a line (i.e., the line passes through the node’s range)?
  – The root is stabbed.
  – All stabbed nodes form a binary tree of depth $\frac{1}{2} \log n$.
  – Hence total number is $O(2^{\frac{1}{2} \log n} = \sqrt{n})$. 

Query a K-d tree
Query a K-d tree

- Complexity:
  - Total time for (1-2): $O(k)$
  - # calls to `Query`: $O(\sqrt{n})$
  - Overall:
    - Report: $O(\sqrt{n} + k)$
    - Count: $O(\sqrt{n})$

```plaintext
Query (t, r) //r: query range
1. If t is a leaf
   1. If t.pt is inside r, return t.pt
   2. Else return NULL
2. If t.range is inside r
   1. ReportTree (t)
3. Else if t.range intersects r
   1. Return Query (t.left, r) ∪ Query (t.right, r)
```
Range Tree

- One binary tree in X (x-tree)
Range Tree

- One binary tree in X \((x\text{-tree})\)
- One binary tree in Y \((y\text{-tree})\)
  for each node in the x-tree
Range Tree

- One binary tree in X (x-tree)
- One binary tree in Y (y-tree)
  for each node in the x-tree
Range Tree

- Space complexity:
  - Size of each tree (x- or y-) is linear to # of leaves
  - Let $T_i$ be # of trees of which $p_i$ is a leaf, total space is
    $\sum_{i=1}^{n} O(\log n)$
  - Total space is $O(n \log n)$
Range Tree

How to build it?
Range Tree

- If t is a node of x-tree:
  - t.val: cut value
  - t.left, t.right: child
  - t.ytree: y-tree
- If t is a leaf of x-tree:
  - t.pt: point
  - t.ytree: a y-tree with a single point

\[ T(n) = O(n) + 2T(n/2) = O(n \log n) \]

BuildXTree (S)  //S: point set
1. If |S|=1, return leaf t where
   1. t.pt and t.ytree are the point of S
2. x be median of X coordinates of all points in S
3. L (R) be subset of S whose X coordinates are no greater than (greater than) x
4. Return node t where
   1. t.val = x
   2. t.left = BuildXTree (L)
   3. t.right = BuildXTree (R)
   4. t.ytree = MergeYTree (t.left.ytree, t.right.ytree)
Range Tree

- Space complexity: $O(n \log n)$
- Building time: $O(n \log n)$
Query a range Tree

Query \( (t, r_X, r_Y) \)

//r_X, r_Y: query range in X and Y
1. If \( t \) is a leaf
   1. If \( t.pt \) is inside \( \{r_X, r_Y\} \), return \( t.pt \)
   2. Else return NULL
2. If \( t.range \) is inside \( r_X \)
   1. QueryY \( (t.ytree, r_Y) \)
3. Else if \( t.range \) intersects \( r_X \)
   1. Return \( \text{Query} (t.left, r_X, r_Y) \cup \text{Query} (t.right, r_X, r_Y) \)

Complexity of QueryY(): \( O(\log n_t + k_t) \)
# Query() calls: \( O(\log n) \)
Total complexity: \( O(\log^2 n + k) \)

Can be improved to \( O(\log n + k) \)
(using fractional cascading, see book/note)