CS133
Computational Geometry
Search Problems
Range Search Problem

Given a set $P$ of points and a rectangular query range $q$, find all the points in $P$ that are enclosed in $q$
Range Search Applications

- Google Maps: Find restaurants in the visible window

- Database applications: Find all employees in the age range [25,35] with salary [80,000, 150,000]
Naïve Range Search

- Scan all the points and compare to \( q \)
- Running time = \( O(n) \)
- Is this optimal?
- Can we do better?
Single-dimensional Ranges

Find all elements in the range [15,55]
Single-dimensional Ranges

Find all elements in the range $[15, 55]$
Single-dimensional Ranges

Find all elements in the range $[15, 55]$
Single-dimensional Ranges

Find all elements in the range [15,55]
Single-dimensional Ranges

Find all elements in the range $[15, 55]$
Single-dimensional Ranges

Find all elements in the range $[15, 55]$
Single-dimensional Ranges

Find all elements in the range $[15, 55]$
Single-dimensional Ranges

Find all elements in the range [15, 55]
Single-dimensional Ranges

Find all elements in the range $[15, 55]$
Single-dimensional Ranges

Find all elements in the range [15, 55]
Single-dimensional Ranges

Find all elements in the range \([15, 55]\)
Single-dimensional Ranges

Find all elements in the range $[15, 55]$
**k-d tree**

- A tree index for a set of k-dimensional points
- Extends BST to k-dimensions
- Each node \( n_i \) stores a point and splits the other points into two subsets
K-d tree Structure
Data Structure

- **Point** = Float[k]
- **Node** {
  - Float key[k];
  - Node *left, *right;
}
- **Rectangle** {
  - Float min[k];
  - Float max[k];
}

K-d tree Construction

1. Build Kd Tree($P$, level)
   1.1. if $|P| = 0$ then return null
   1.2. $a = level \% k$
   1.3. Find the median $p_m$ of $P$ along the axis $a$
   1.4. left = Build Kd Tree($\{p \in P | p[a] \leq p_m[a]\}$, levels+1)
   1.5. right = Build Kd Tree($\{p \in P | p[a] > p_m[a]\}$, levels+1)
   1.6. Return New Node($p_m$, left, right)

2. Initial call:
   2.1. Build Kd Tree($P$, 0)
Range Search

- Search(node, q, level)
  - if q contains(node.value)
    - Report node.value
  - \( a = level \% k \)
  - if node.value[a] ≤ q.max[a]
    - Search(node.left, q, level + 1)
  - if node.value[a] ≥ q.min[a]
    - Search(node.right, q, level + 1)