Universiteit Antwerpen
Faculteit Toegepaste Ingenieurswetenschappen

Qaudtree

Data Structure – Thomas Kramp

Quadtree

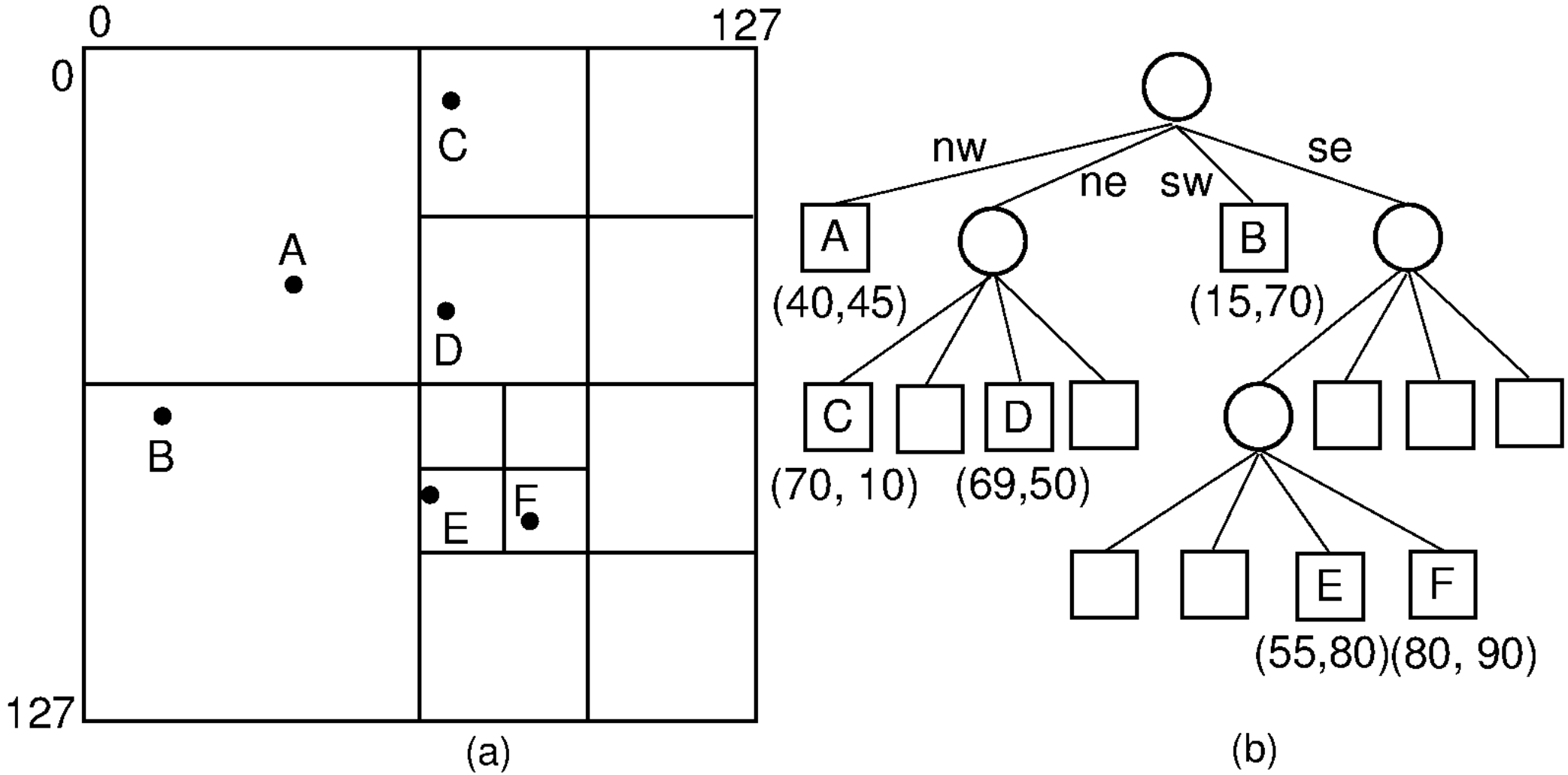
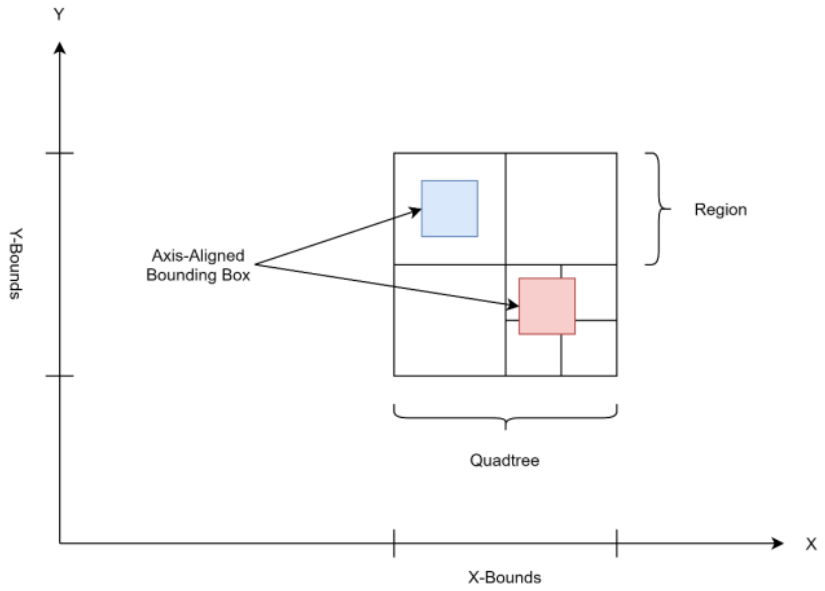


Image Quadtree: https://opendsa-server.cs.vt.edu/ODSA/Books/Everything/html/PRquadtree.html

Requirements

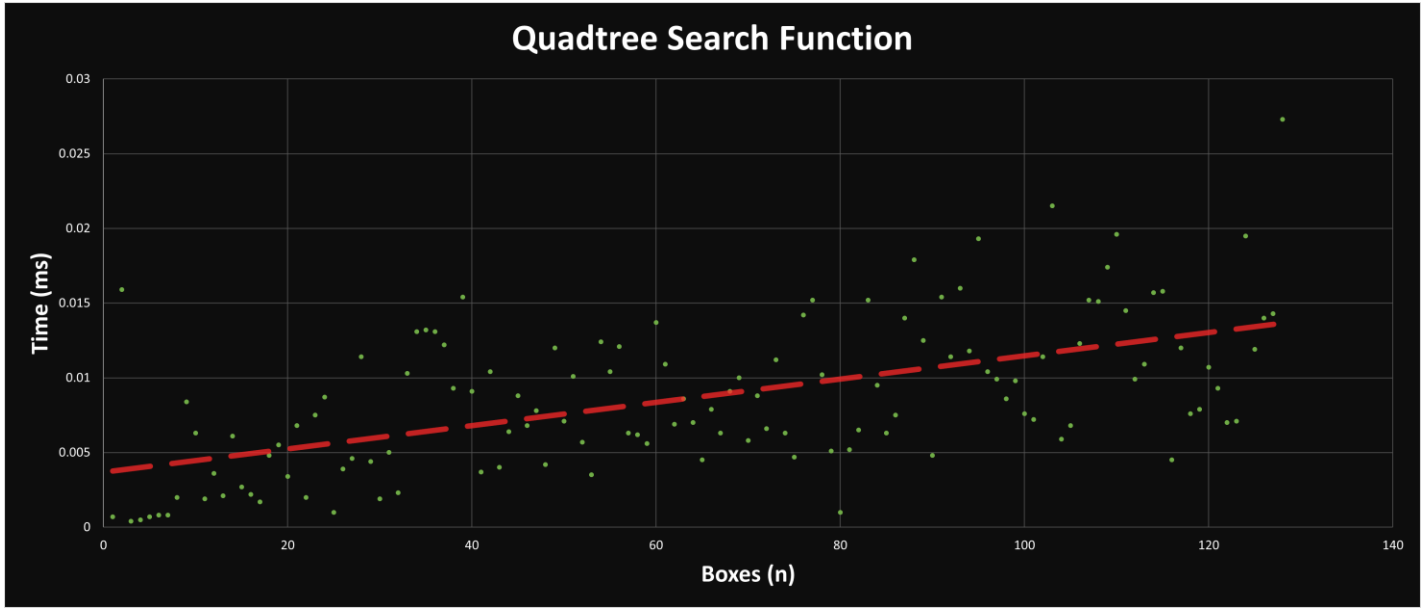
- AxisAlignedBoundingBox → AxisAlignedBoundingBox & MetaBoundingBox
- Quadtree → Quadtree
  - Insert → Quadtree & AxisAlignedBoundingBox
  - Search → MetaBoundingBox
  - Begin & End → MetaBoundingBox & QuadtreeIterator

- MetaBoundingBox → Hash
- QuadtreeIterator → Operators

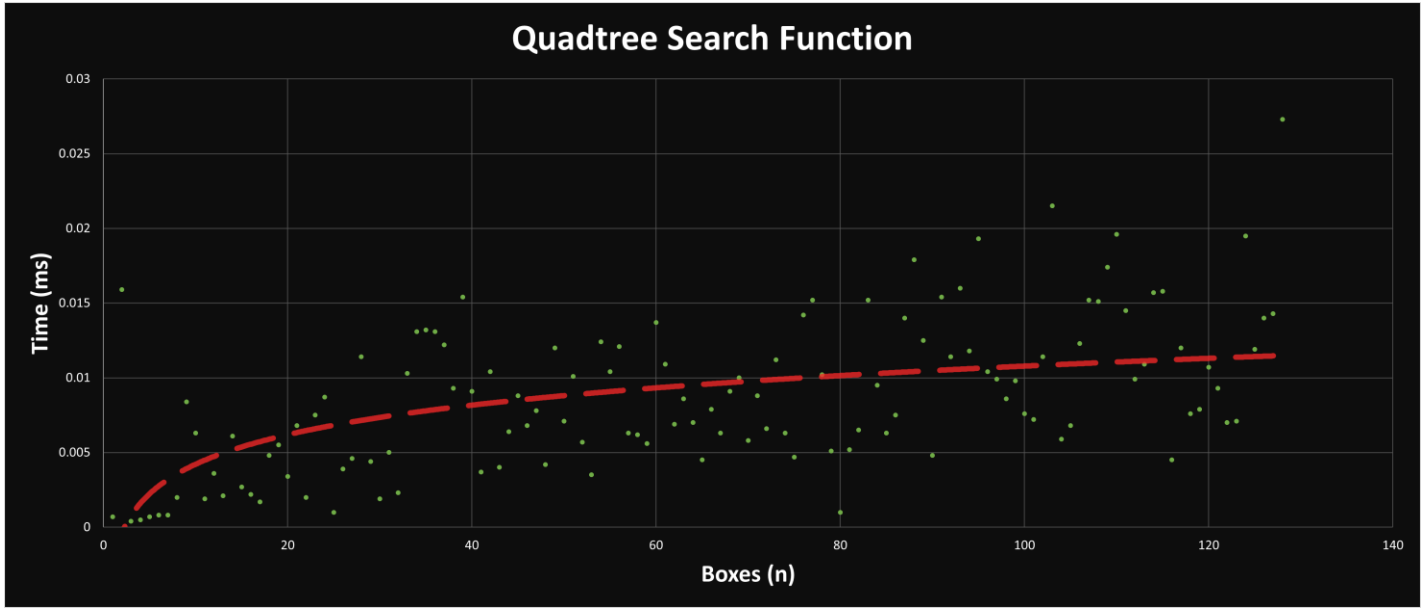


Graphs (n = 128)

Linear O(n)

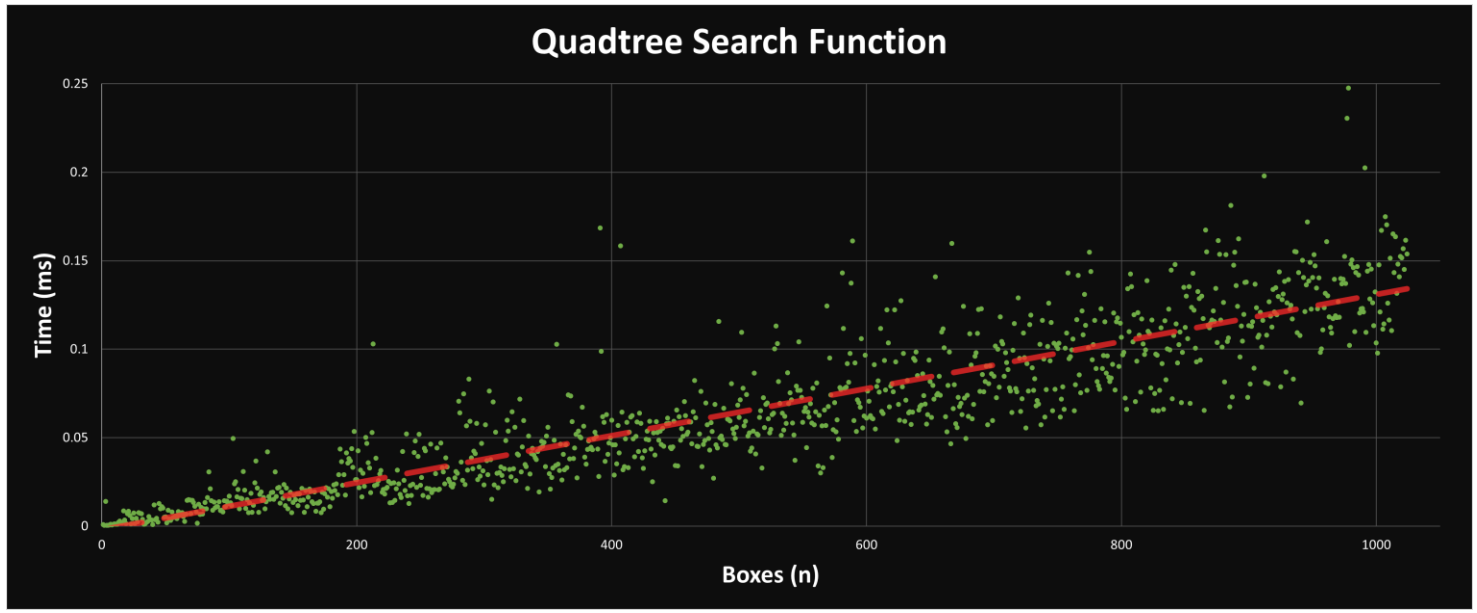


Logaritmic O(log(n))

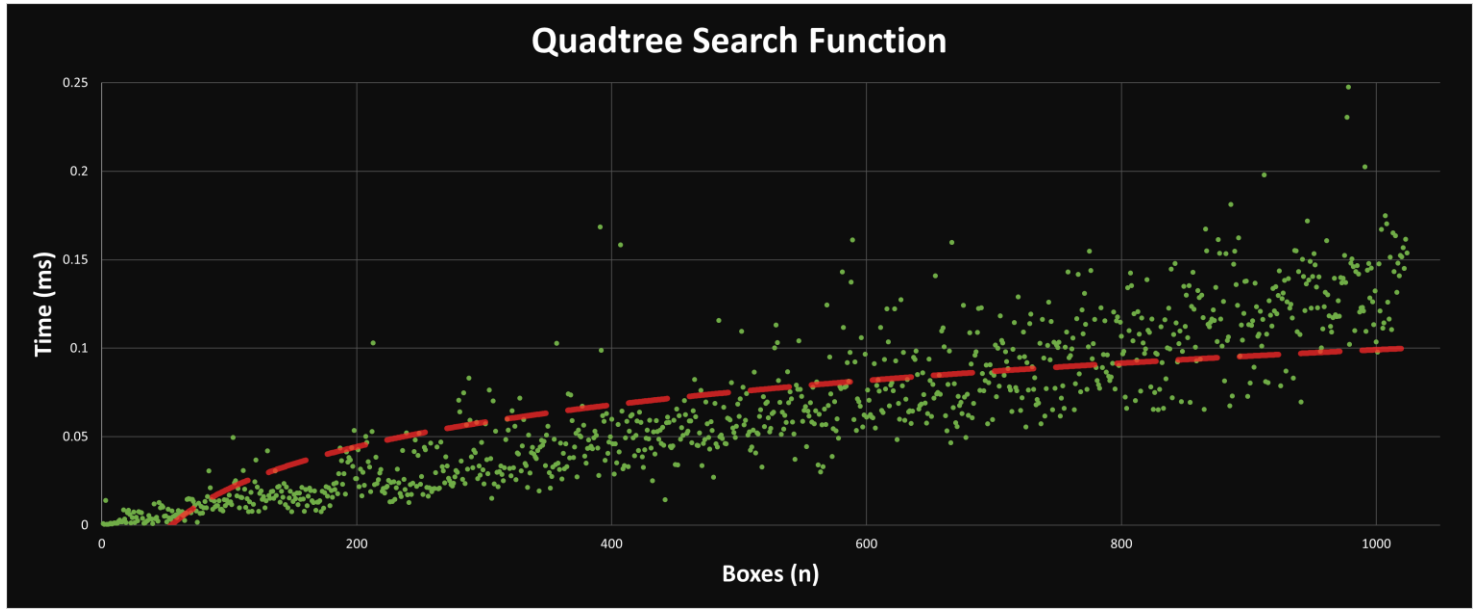


Graphs (n = 1024)

Linear O(n)

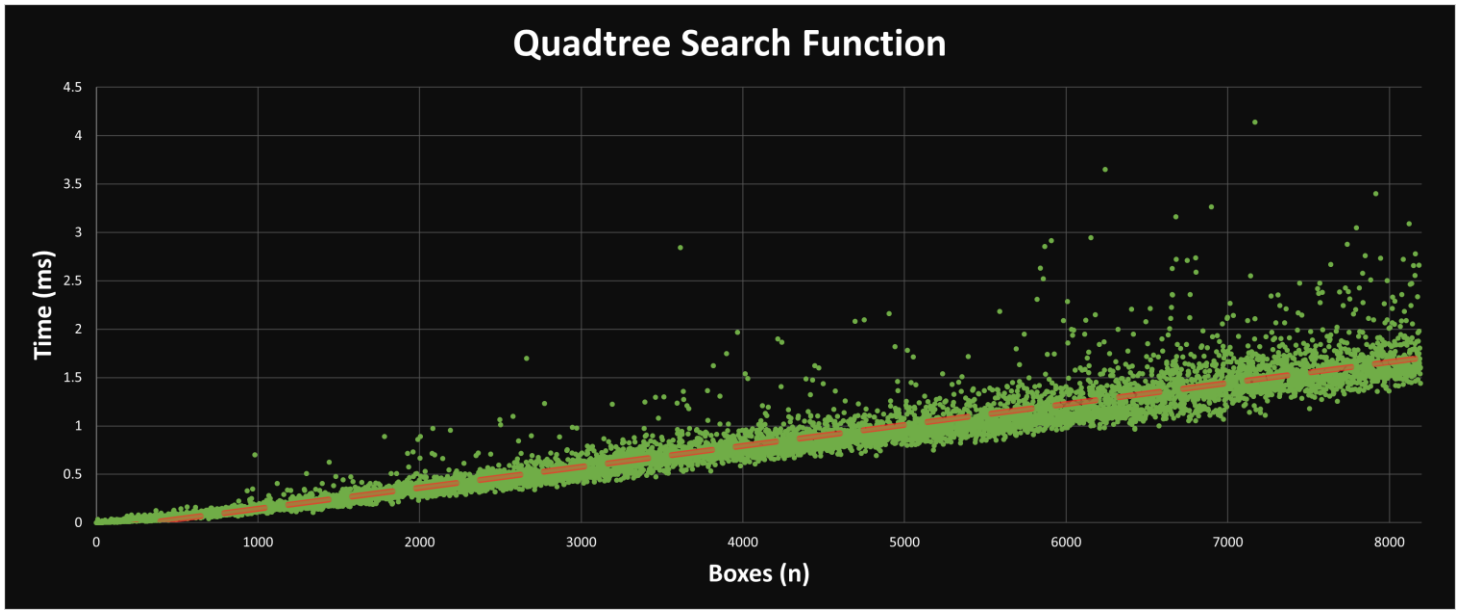


Logaritmic O(log(n))

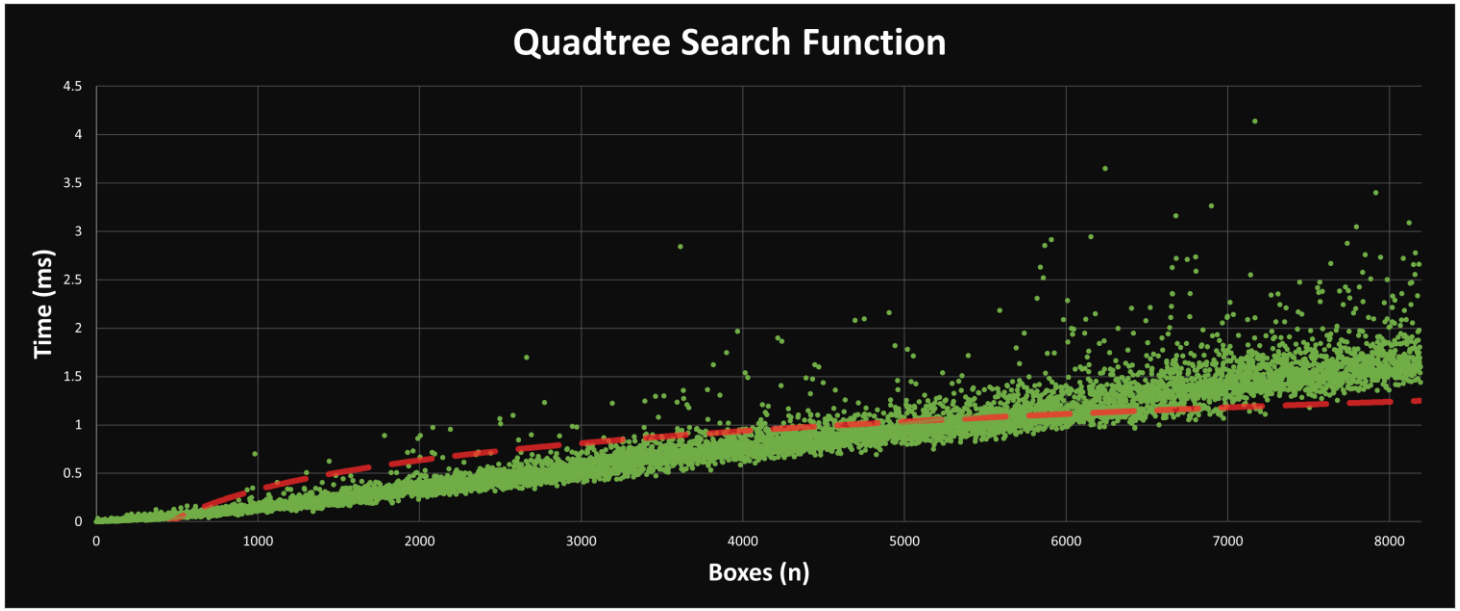


Graphs (n = 8192)

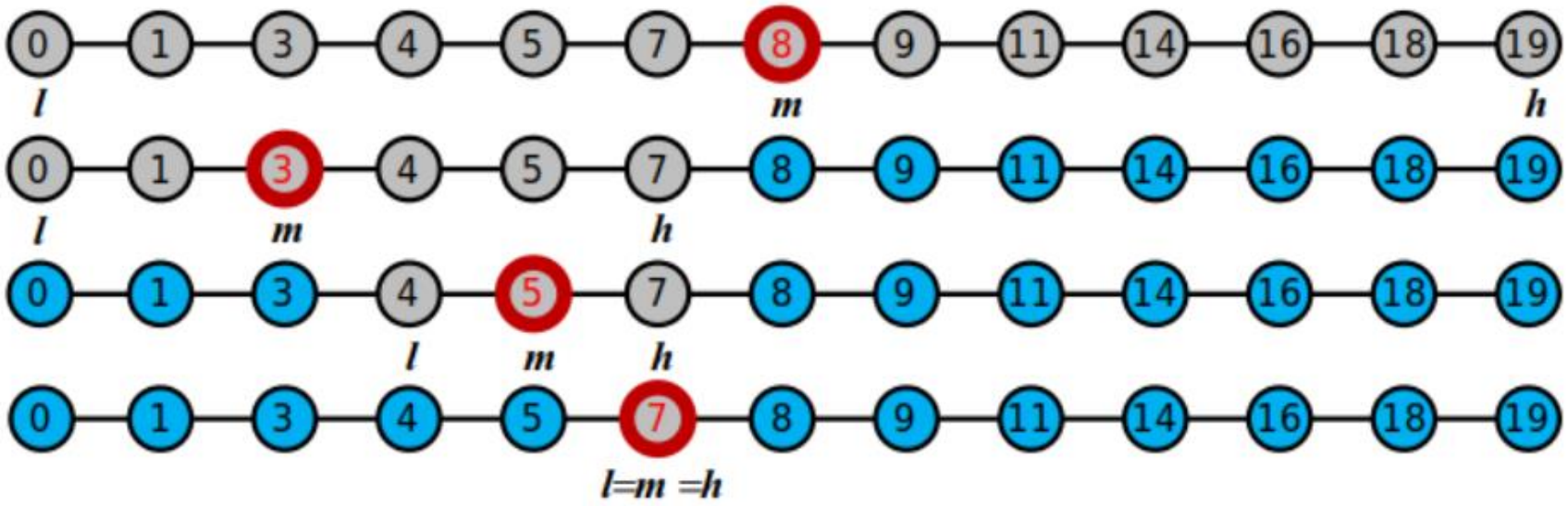
Linear O(n)

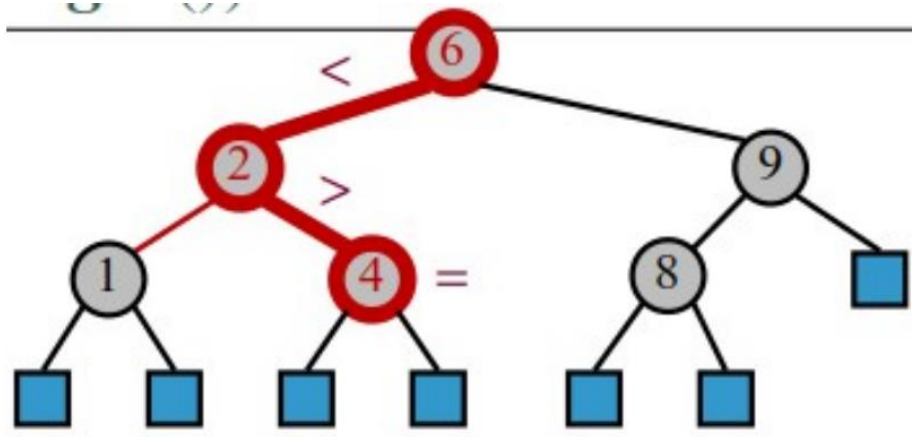


Logaritmic O(log(n))



Binary Search → O(log(n))





Code (Stucture)

```cpp
template<typename Metadata>
std::unordered_set<MetaBoundingBox<Metadata>> Quadtree<Metadata>
        ::query_region(const AxisAlignedBoundingBox &container) {
    std::unordered_set<MetaBoundingBox<Metadata>> boxes = std::unordered_set<MetaBoundingBox<Metadata>>();
    if (subZones.empty()) {
        // If there aren't any sub-zones loop through all boxes
        for (auto &abBox: abBoxes) {
            // Add the box if there is a collision
            if (collides(abBox.getBox(), container)) boxes.insert(abBox);
        }
    } else {
        // If there are sub-zones loop through all sub-zones
        for (auto &zone: subZones) {
            // Look into sub-zone if there is a collision
            if (collides(zone.getBounds(), container)) {
                // Get all colliding boxes in sub-zone
                std::unordered_set<MetaBoundingBox<Metadata>> zone_boxes = zone.query_region(container);
                // Add all the colliding boxes
                boxes.insert(zone_boxes.begin(), zone_boxes.end());
            }
        }
    }
    return boxes;
}
```

Leaf

Node

Code (Loop)

```cpp
template<typename Metadata>
std::unordered_set<MetaBoundingBox<Metadata>> Quadtree<Metadata>
        ::query_region(const AxisAlignedBoundingBox &container) {
    std::unordered_set<MetaBoundingBox<Metadata>> boxes = std::unordered_set<MetaBoundingBox<Metadata>>();
    if (subZones.empty()) {
        // If there aren't any sub-zones loop through all boxes
        for (auto &abBox: abBoxes) {
            // Add the box if there is a collision
            if (collides(abBox.getBox(), container)) boxes.insert(abBox);
        }
    } else {
        // If there are sub-zones loop through all sub-zones
        for (auto &zone: subZones) {
            // Look into sub-zone if there is a collision
            if (collides(zone.getBounds(), container)) {
                // Get all colliding boxes in sub-zone
                std::unordered_set<MetaBoundingBox<Metadata>> zone_boxes = zone.query_region(container);
                // Add all the colliding boxes
                boxes.insert(zone_boxes.begin(), zone_boxes.end());
            }
        }
    }
    return boxes;
}
```

Loop Boxes

Loop Zones

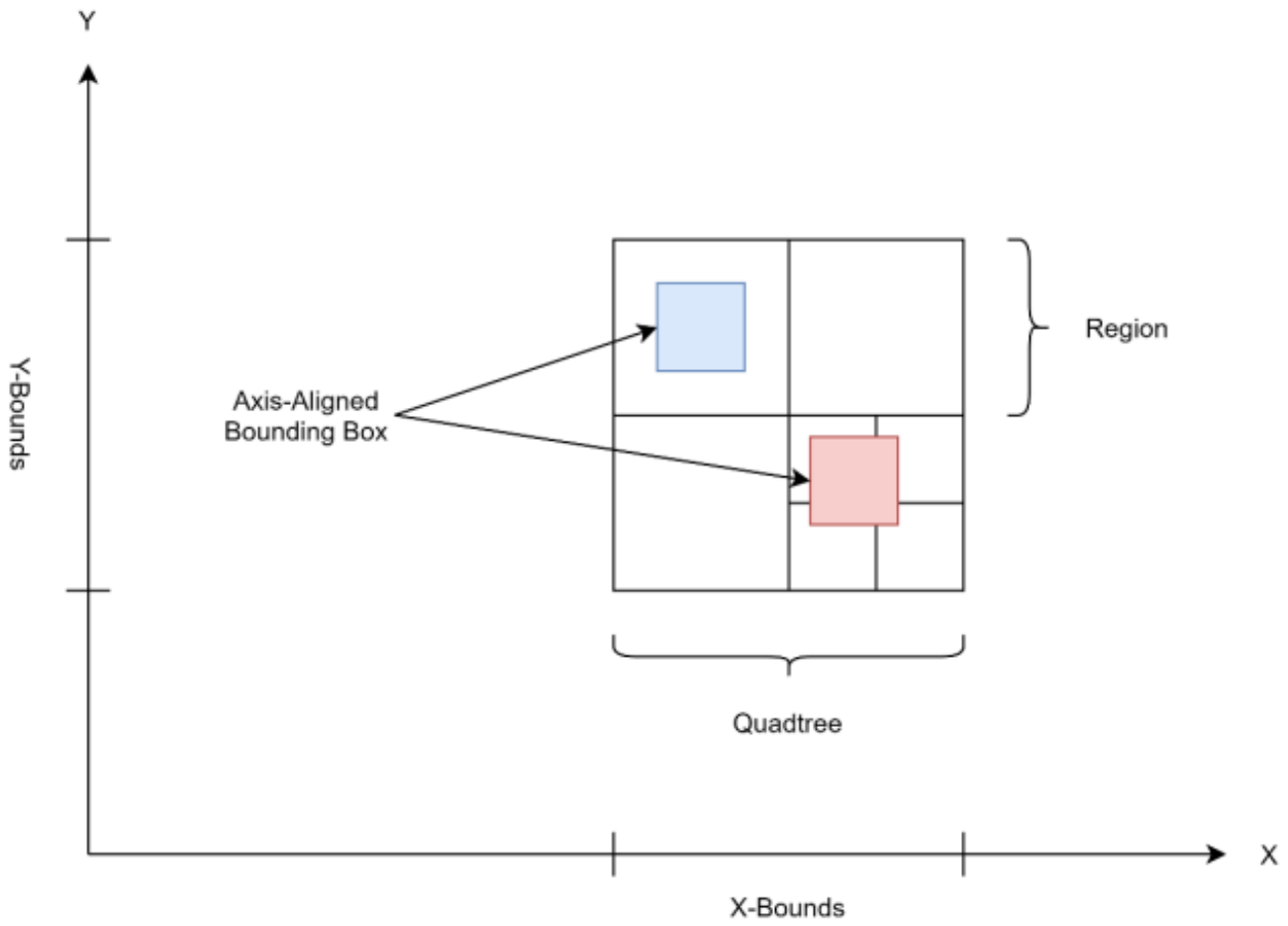
Code (Check)

```cpp
template<typename Metadata>
std::unordered_set<MetaBoundingBox<Metadata>> Quadtree<Metadata>
        ::query_region(const AxisAlignedBoundingBox &container) {
    std::unordered_set<MetaBoundingBox<Metadata>> boxes = std::unordered_set<MetaBoundingBox<Metadata>>();
    if (subZones.empty()) {
        // If there aren't any sub-zones loop through all boxes
        for (auto &abBox: abBoxes) {
            // Add the box if there is a collision
            if (collides(abBox.getBox(), container)) boxes.insert(abBox);
        }
    } else {
        // If there are sub-zones loop through all sub-zones
        for (auto &zone: subZones) {
            // Look into sub-zone if there is a collision
            if (collides(zone.getBounds(), container)) {
                // Get all colliding boxes in sub-zone
                std::unordered_set<MetaBoundingBox<Metadata>> zone_boxes = zone.query_region(container);
                // Add all the colliding boxes
                boxes.insert(zone_boxes.begin(), zone_boxes.end());
            }
        }
    }
    return boxes;
}
```

Check Box

Check Zone

Quadtree


Y
Y-Bounds
Axis-Aligned
Bounding Box
Region
Quadtree
X
X-Bounds

Other Reasons

- To file doens’t work → To command output
- Clion has been been difficult on windows
- Boxes in multiple trees
- Collision detection
- Double hashing

```
uadTree.insert(AxisAlignedBoundingBox(1, 2, 3, 4), "figure1");
uadTree.insert(AxisAlignedBoundingBox(5, 6, 7, 8), "figure2");
```

- Should return { figure1, figure2 }, but returns { figure2, figure1 }