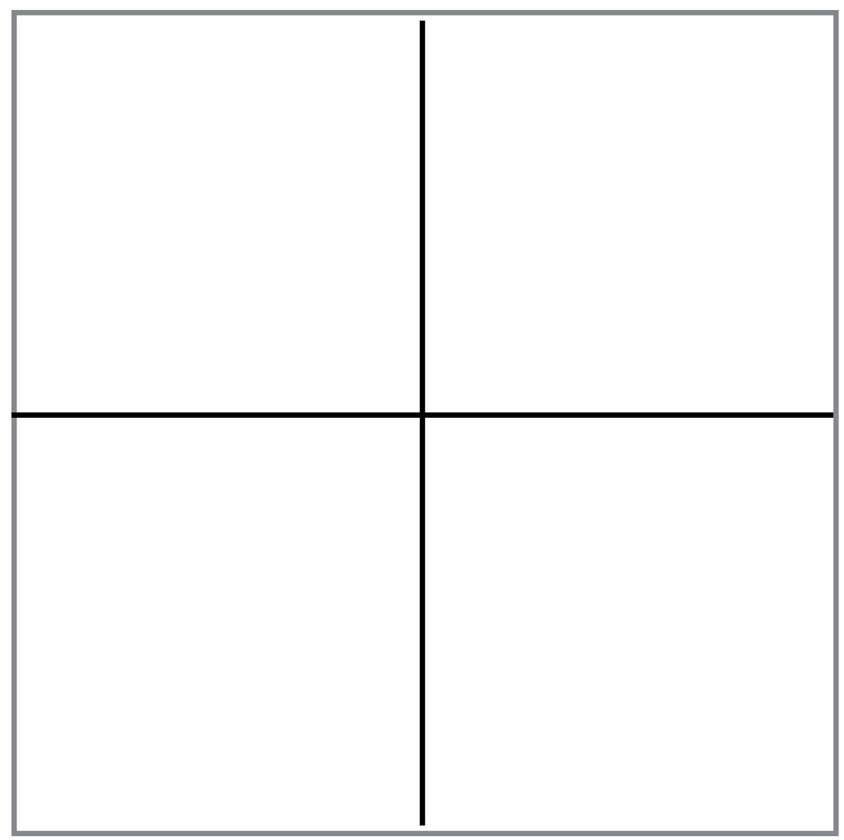
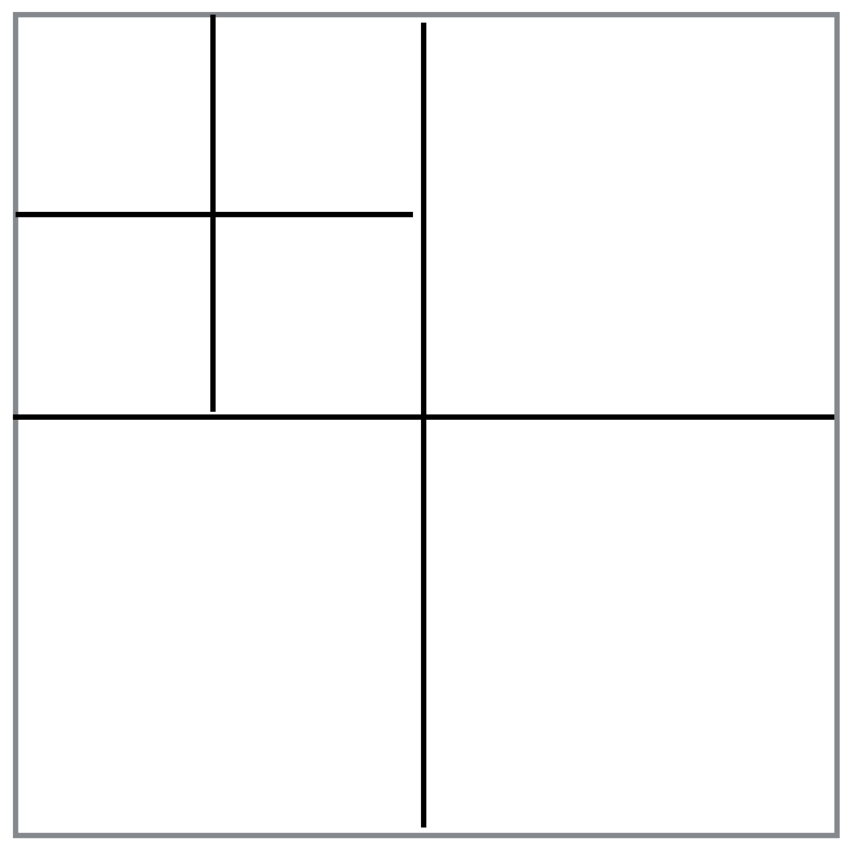
Algorithms for GIS:

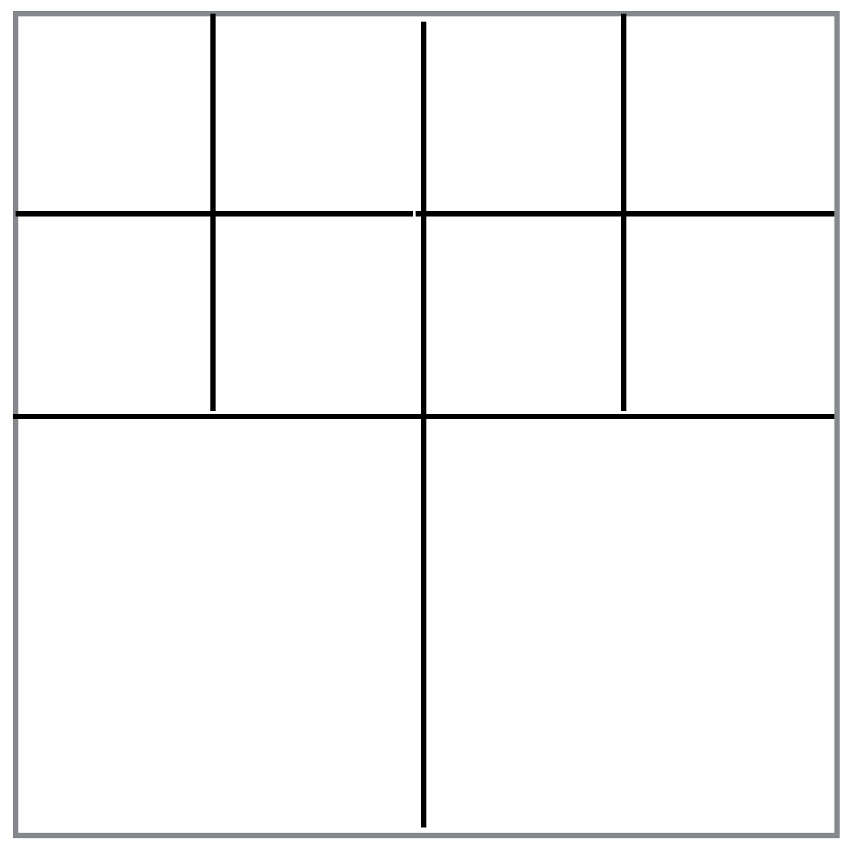
Quadtrees

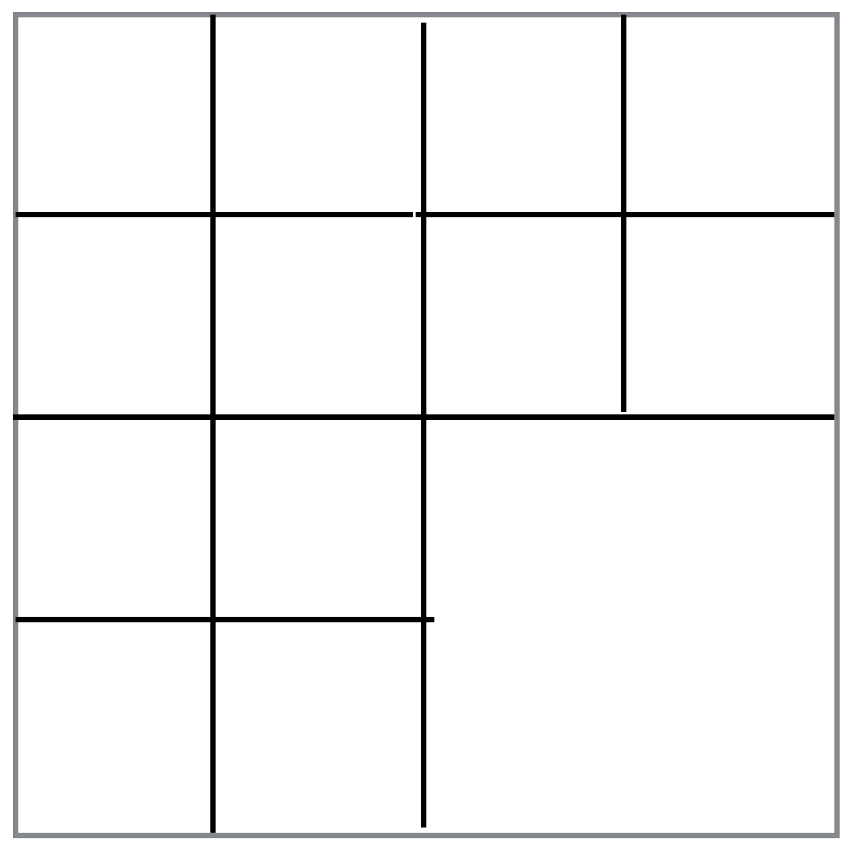
Quadtree

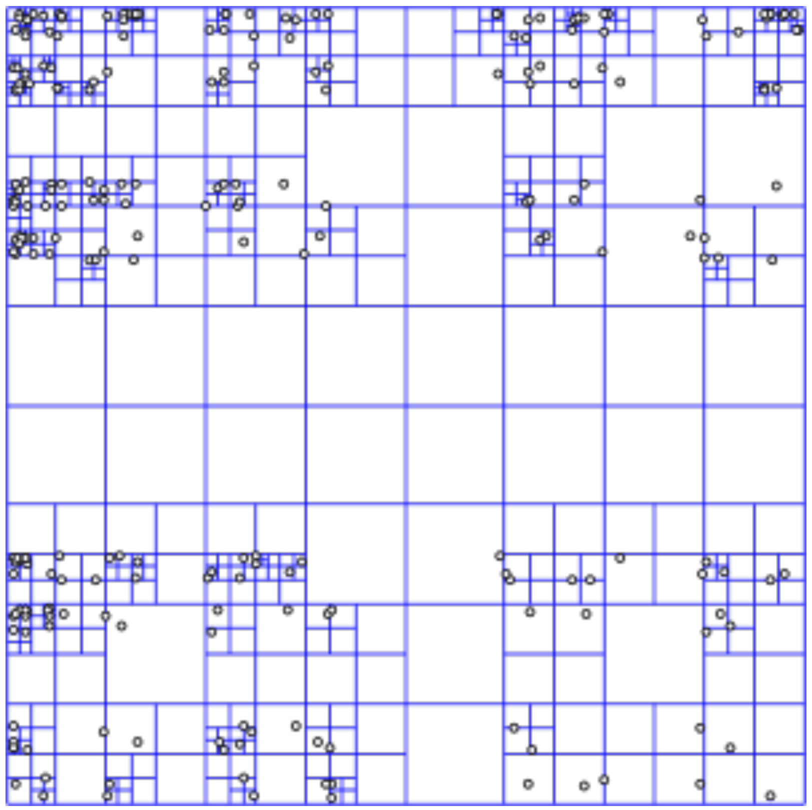
- A data structure that corresponds to a hierarchical subdivision of the plane
- Start with a square (containing inside input data)
    - Divide into 4 equal squares (quadrants)
    - Continue subdividing each quadrant recursively
    - Subdivide a square until it satisfies a stopping condition:
        - a quadrant is “small” enough, for e.g. contains at most 1 point











Quadtrees

- Simple and versatile data structure
- Lots of applications
- Quadtree can be built for
    - points
    - edges
    - polygons
    - images
- Generalizes to d dimensions
    - d=3: octree
- Many variants of quadtrees have been proposed
- Hundreds of papers

Point-quadtree

Let P = set of n points in the plane

- Input: P
- Problem: Store P in a quadtree
    - such that every square has <= 1 point

- Questions:

1. Size?
2. How to build it and how fast?
3. What can we do with it?

Exercises

Let P = set of n points in the plane

- Draw the quadtree corresponding to a regular grid
  - how many nodes does it have?
  - how many leaves?
  - height?

- Pick a set of points with a non-uniform distribution and draw the quadtree
  - how many nodes does it have?
  - how many leaves?
  - height?

Exercises

- Let n=2 ==> we'll look at sets of 2 points in the plane.
  - Sketch the smallest possible quad tree for two points in the plane.
  - Sketch the largest possible quad tree for two points in the plane.
  - Give an upper bound for the height of a quadtree for 2 points.

Size

Let P = set of n points in the plane

Theorem: The height of a quadtree storing P is at most $\lg (s/d) + 3/2$, where s is the side of the original square and d is the distance between the closest pair of points in P.

Proof: Each level divides the side of the quadrant into two. After i levels, the side of the quadrant is s/2^i. …

This means that…

- The distance between points can be arbitrarily small, so the height of a quad tree can be arbitrarily large in the worst case.

Building a quad tree

Let P = set of n points in the plane

Node buildQuadtree(set of points P, square S)

- if P has at most one point:
  - build a leaf node , store P in it, and return node
- else
  - partition S into 4 quadrants S1, S2, S3, S4
  - partition P into P1, P2, P3, P4
  - create a node
  - node ->child1 = buildQuadtree(P1, S1)
  - node ->child2 = buildQuadtree(P2, S2)
  - node ->child3 = buildQuadtree(P3, S3)
  - node ->child4 = buildQuadtree(P4, S4)
  - return node

Building a quadtree

Let P = set of n points in the plane

- How long does it take?
- Let the height of the quadtree be h.

- Analysis:
  - Total time = total time in partitioning + total time in recursion
  - Partitioning
    - Partitioning P into P1, P2, P3, P4 runs in time O(|P|).
    - We cannot bound precisely each P1, P2, P3, P4 (each can have anywhere between 0 points and n points);
    - But if we look at all nodes at same level in the quadtree: together they partition the input square and all their sets add up to precisely n
    - The time to partition, summed over the entire quadtree, will be O(n) per level, or O(h x n) in total
  - Recursion:
    - Every recursive call creates a node
    - How many nodes?

Building a quad tree

Let P = set of n points in the plane

Question: What is the total size (number of nodes) in a quadtree if height h, storing n points?

- Quadtree consists of leaves and internal nodes
  - Let L_i = number of leaves
  - Let N_i = number of internal nodes
- Counting leaves: each node has 0 or 4 children.
  - Claim: The number of leaves is 1 + 3 N_i
- Counting the internal nodes:
  - Each internal node has at least two point inside it (otherwise it would satisfy the stopping criterion and would be a leaf)
  - At each level of the quadtree: the internal nodes define a partition of the input square ==> O(n) nodes per level
  - Total N_i = h x O(n)
- Total:
  - L_i+N_i = (1+3N_i) + N_i = O(N_i) = O(h x n)

Building a quad tree

Let P = set of n points in the plane

Theorem: A quadtree for P can be built in $O(h \times n)$ time, where h is the depth (height) of the quad tree.

Building a quad tree

Let P = set of n points in the plane

Theorem: A quadtree for P can be built in $O(h \times n)$ time, where h is the depth (height) of the quad tree.

Point quadtree: Summary

Let P = set of n points in the plane

- A quadtree for P has height O(lg (1/d)).
- A point quadtree of n points can be built in O(h x n) time.

- Theoretical worst case:
  - height is unbounded in the worst case
- In practice:
  - often h=O(n) and build time is $O(n^2)$ Note:
  - For sets of points that are approximately uniformly distributed, we have that h = O(lg n) and the running time becomes O(n lg n).
  - In many practical situations quad trees have logarithmic height and can be built in O(n lg n) time

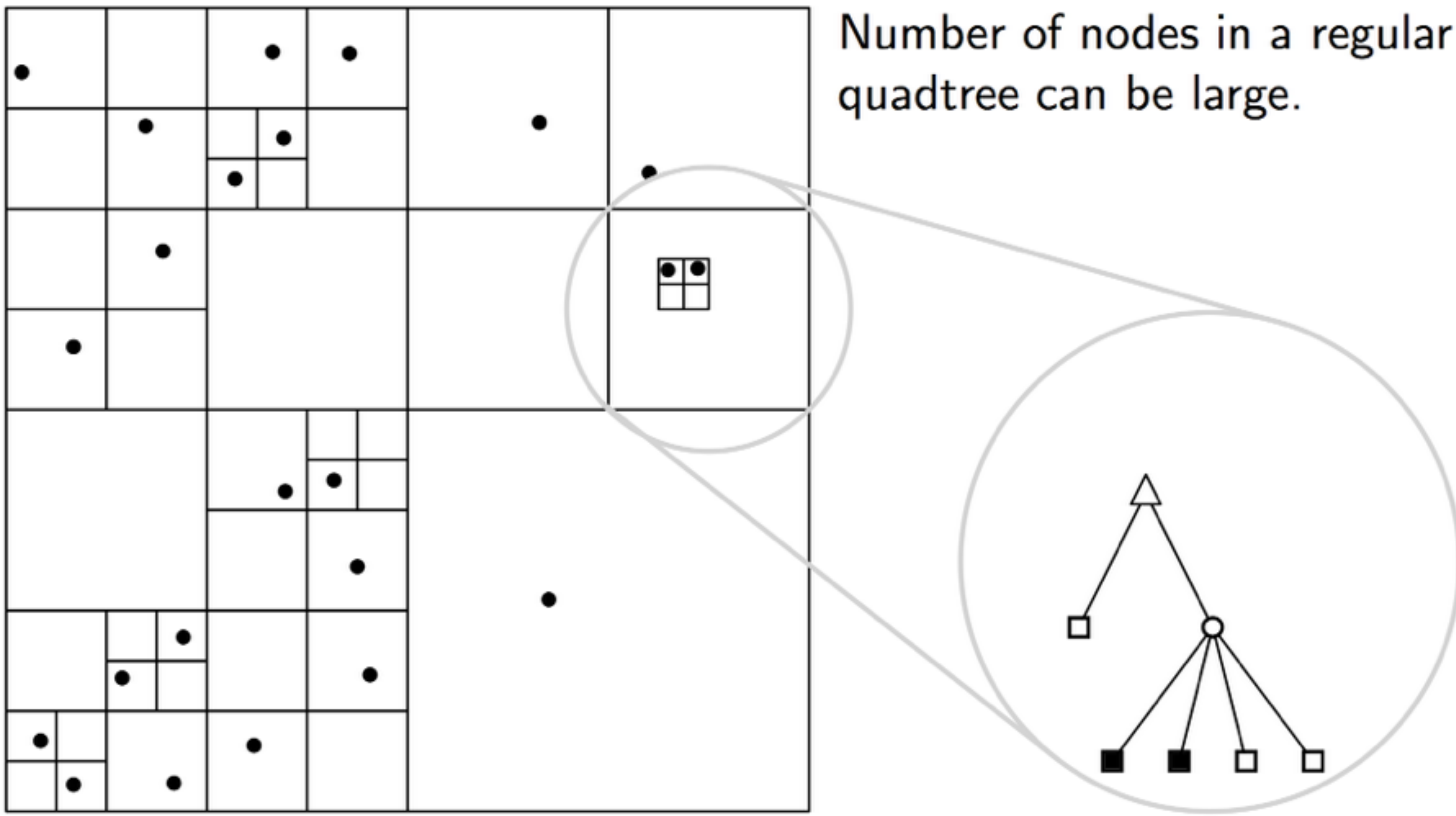
Point quadtree: Summary

Let P = set of n points in the plane

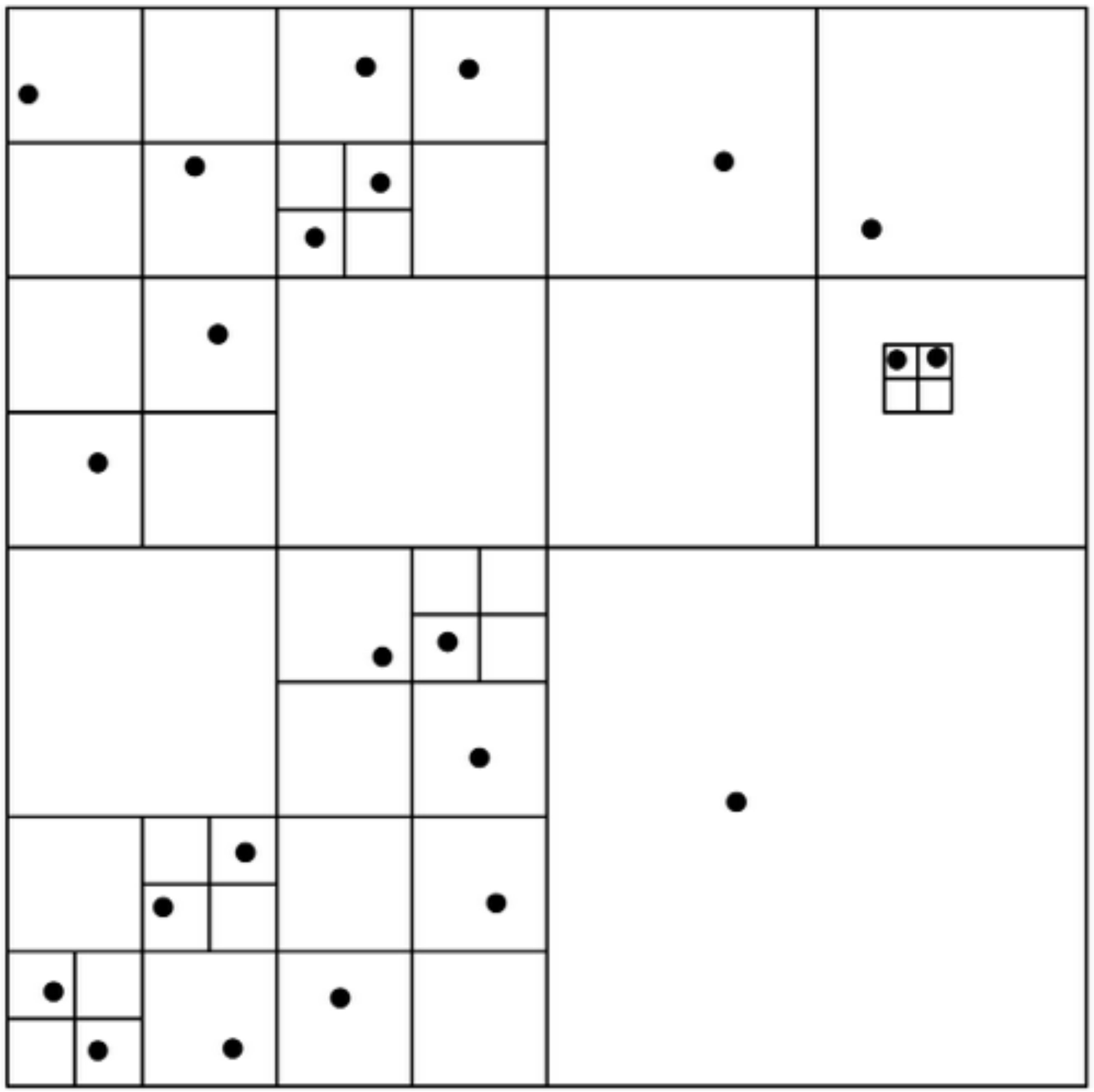
- A quadtree for P has height O(lg (1/d)).
- A point quadtree of n points can be built in O(h x n) time.

- Extensions:
  - compressed quadtrees: height is h=O(n) **in the worst-case**
    - idea: compress paths of nodes with 3 empty children into one node, called a *donut*

==> a node may have 5 children, an empty *donut* + 4 regular quadrants

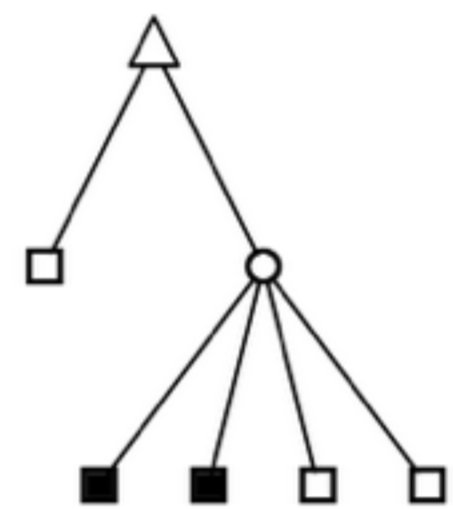


Number of nodes in a regular quadtree can be large.



Number of nodes in a regular quadtree can be large.

Number of nodes in a compressed quadtree is $O(n)$.



Applications of quadtrees

- Hundreds of papers..
- Specialized quadtrees for storing edges (edge quad trees) , polygons, etc
- Used to answer queries on spatial data such as:
  - find the nearest neighbor (NN) of this point
  - find the k-NNs of this point
  - find all points in this rectangle (range searching)
  - find all segments intersecting a given segment
  - etc

Applications of quadtrees

- Used for fast rendering (LOD)
  - Level i in the qdt —> scene at a certain resolution
  - bottom level has full resolution
  - render scene at a resolution dependent on its distance from the viewpoint

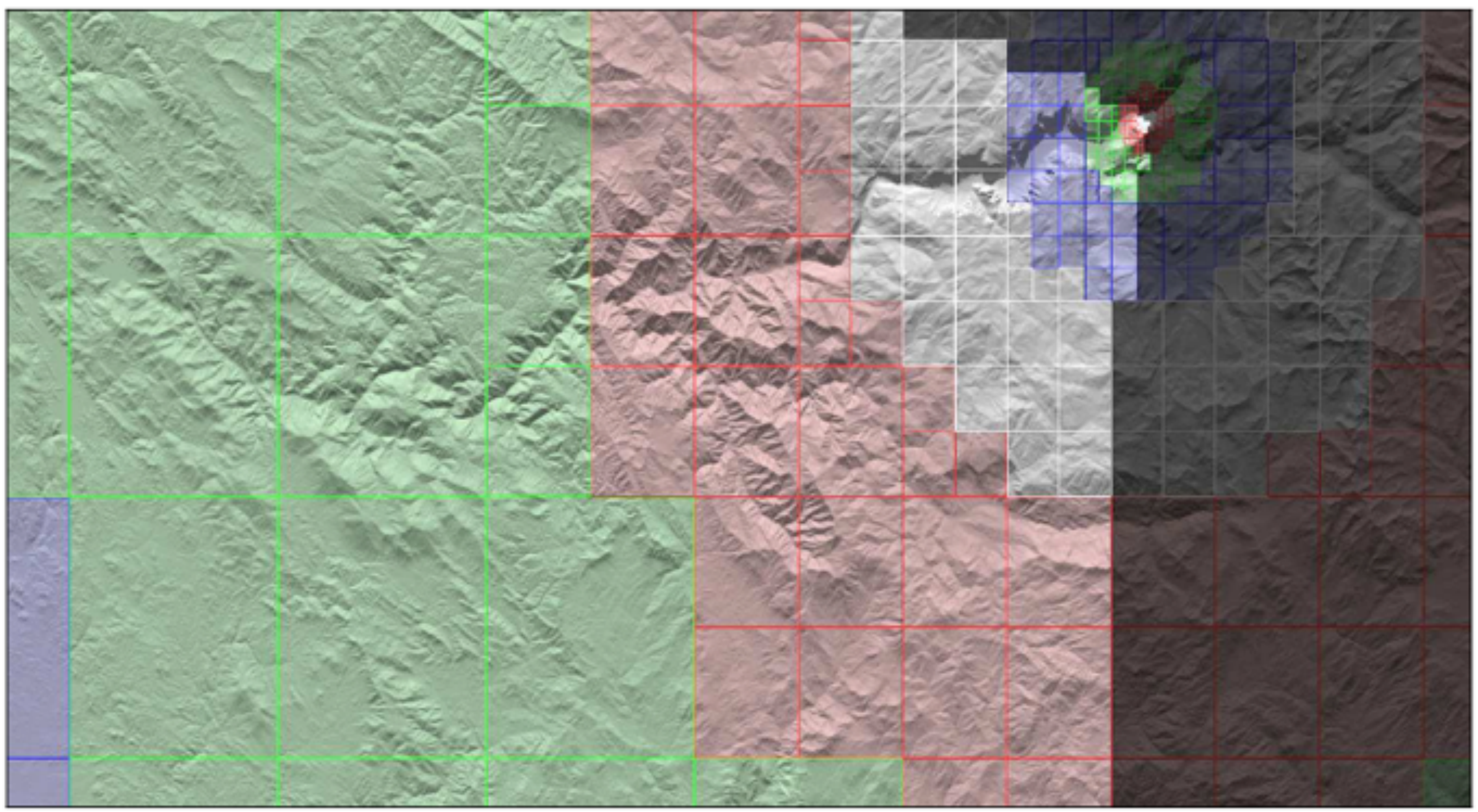


***Figure 3** LOD selection of quadtree nodes (the frustum culled section is shaded in dark).*

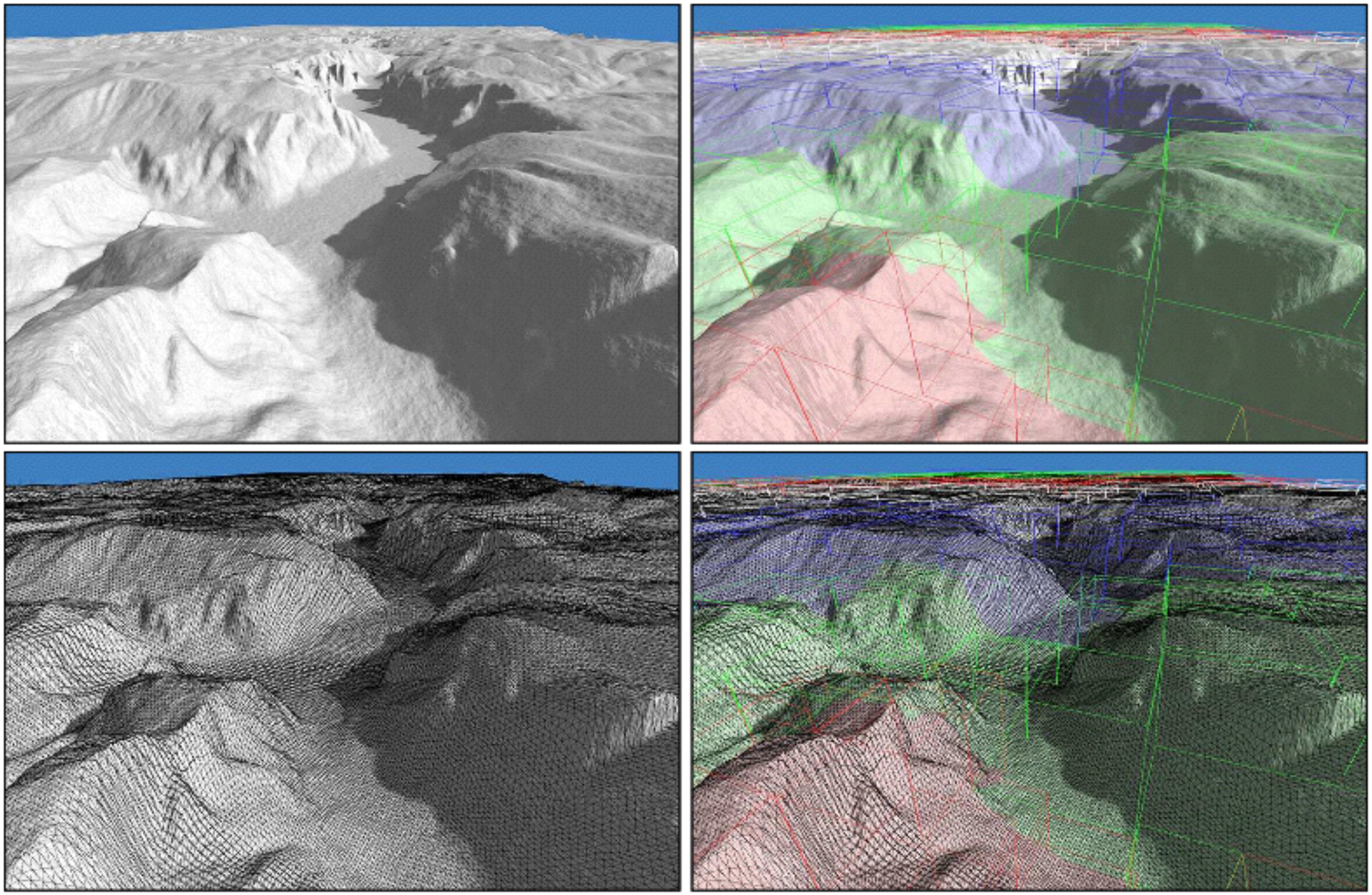


Figure 5 *Distribution of LOD levels and nodes (different colors represent different layers).*

Applications of quadtrees

- Image analysis/compression