Binary Space Partitions (chapter 12)

Elisha Sacks
Range queries apply to geometric elements beyond points.

The standard approach is to decompose the domain into regions that intersect a constant number of elements.

The regions that intersect a query box $Q$ are enumerated and the elements in each region that intersect $Q$ are returned.

Another application: find all the intersecting pairs of elements.

kd-trees are a good decomposition for points.

We will see two decompositions for other types of elements.

- BSP trees use convex regions (this lecture).
- Quadtrees use boxes (chapter 14).
A BSP (binary space partition) tree decomposes the plane into convex regions.

An internal node contains a line that splits its region into the regions of its two children.

A leaf contains pointers to the inputs that intersect its region.

Alternately, a leaf contains the input fragments in its region.
The region of the root is the plane.

The region of an internal node is the intersection of the region of its parent with a halfspace of the splitting line of its parent.
Auto-Partitions

A BSP for a set of line segments is an auto-partition if every splitting line is the support line of a segment.

**Algorithm**

If the input size is less than two,
   return a leaf containing the input.
else
   Pick a segment at random and let \( l \) be its support line.
   Split the segments by \( l \) to obtain two sets of fragments.
   (\( l \) is not in either set.)
   Recursively construct the child BSP trees.
Influence of Insertion Order

- The number of splits depends on the segment insertion order.
- (a) Inserting $s_1$ splits $s_2$ and $s_3$.
- (b) This order has no splits.
Lemma 12.1 The expected number of fragments is \( O(n \log n) \).

Proof

Let \( s_j \) be a segment that intersects the support line \( \ell \) of segment \( s_i \). A segment separates \( s_i \) and \( s_j \) when it intersects \( \ell \) between them. Denote the number of intersecting segments by \( \text{dist}_i(s_j) \). There are at most two \( s_j \) with \( \text{dist}_i(s_j) = k \) for \( 0 \leq k \leq n - 2 \).
Number of Fragments (continued)

For the $s_i$ insertion to split $s_j$, it must occur before $s_j$ and their separating segments are inserted.

This probability is $\frac{1}{\text{dist}_i(s_j)+2}$ because each of these segments is equally likely to be first.

The expected number of splits is bounded by

$$\sum_{j \neq i} \frac{1}{\text{dist}_i(s_j)+2} \leq 2 \sum_{k=0}^{n-2} \frac{1}{k+2} \leq 2 \log n$$

Summing over the $n$ segments yields $n + 2n \log n$ fragments.
Running Time

**Theorem 12.2** The auto-partition algorithm runs in expected time $n^2 \log n$.

**Proof** The number of recursive calls is bounded by the number of fragments, which is $O(n \log n)$ by Lemma 12.1, and the cost of a call is $O(n)$ excluding recursive calls.
Discussion of Complexity

- The auto-partition algorithm constructs a BSP of size $n \log n$ in expected time $n^2 \log n$.
- A BSP of size $n \log n$ can be constructed in time $n \log n$.
- $n$ segments can require a BSP of size $n \log n / \log \log n$.
- $n$ spatial triangles can require a BSP of size $n^2$.
- These lower bounds do not occur for typical inputs.
- We will study a theoretical model of this empirical fact.
Scene Density

The density of a set of objects is the smallest $\lambda$ such that a ball $B$ of diameter $d$ intersects at most $\lambda$ objects of diameter $d$ or greater.

- The figure shows eight line segments with $\lambda = 3$.
- The definition is independent of scale and dimension.
- Density is typically low and independent of input size.
- Designing density sensitive algorithms is worthwhile.
Density Sensitive BSP Construction

- Handle low density shapes of arbitrary complexity.
- Represent an object by the corners of its bounding box.
- The corners, called guards, guide BSP construction.
- Phase 1: construct a quad-BSP tree for the guards.
- Phase 2: insert the objects into this tree.
- Phase 3: refine the leaves with a BSP algorithm.
Lemma 12.6 An axis-parallel square that contains $k$ guards intersects at most $k + 4\lambda$ objects.

Proof

- Let $\sigma$ be an axis-parallel square containing $k$ guards.
- At most $k$ objects have a guard inside $\sigma$.
- The guardless objects $S'$ have density at most $\lambda$.
- We will show that at most $4\lambda$ objects in $S'$ intersect $\sigma$. 
Guard Lemma (continued)

- The $x$-extent or the $y$-extent of any $o \in S'$ contains that of $\sigma$.
- The diameter of $o$ exceeds the edge size of $\sigma$, so $d(o) \geq d(\sigma)/\sqrt{2} > d(\sigma)/2$.
- Cover $\sigma$ with four disks of diameter $d(\sigma)/2$.
- $o$ intersects at least one disk.
- Each disk intersects at most $\lambda$ objects in $S'$ because their diameters exceed that of the disks.
A quad-BSP tree is a BSP tree with axis parallel split lines.

We construct a quad-BSP tree with at most $k$ guards per leaf.

The root region is a box that contains the input objects.

The figure shows a $k = 1$ tree in which each region is split into equal pieces along its longer dimension.

This is the standard quadtree construction algorithm.
Quadtree Split

A quadtree split divides a rectangle into four congruent rectangles by means of three BSP splits.

Quadtree splits yield overly large trees when guards are clustered.
An adaptive split strategy yields $O(n/k)$ leaf nodes.

1. Perform an initial quadtree split.
2. If exactly one of the four rectangles $\sigma$ contains over $k$ guards
   2.1 Reject the quadtree split.
   2.2 Shrink $\sigma$ along its diagonal until $\sigma'$ excludes $k$ guards.

The examples have $k = 4$. 
Lemma 12.7 Phase 1 produces a BSP tree with $O(n/k)$ leaf nodes each of which intersects at most $k + 4\lambda$ objects.

Proof

- Let $f(m)$ be the number of internal nodes for $m$ guards.
- There are $f(m) + 1$ leaf nodes.
- If $m \leq k$, the tree is a leaf and $f(m) = 0$.
- If $m > k$, three internal nodes create four rectangles.
- Let the $i$th rectangle contain $m_i$ guards.
- Let $J = \{i : 1 \leq i \leq 4 \text{ and } m_i > k\}$.
- A guard is in at most one rectangle, so $\sum_{i \in J} m_i \leq m$.
- $f(m) = 3 + \sum_{i \in J} f(m_i)$. 
We will prove that \( f(m) \leq 6m/k - 3 \) by induction.

The BSP tree has \( O(n/k) \) leafs because \( n \leq 4m \).

The base case is \( m \leq k \), so assume \( m > k \).

If \( J \) is empty, \( f(m) = 3 \leq 6m/k - 3 \) because \( m > k \).

If \( J \) has one member, an adaptive split occurs, that member contains at most \( m - k \) guards, and
\[
  f(m) \leq 3 + f(m - k) \leq 3 + 6(m - k)/k - 3 = 6m/k - 3.
\]

If \( J \) has two or more members, \( m_i < m \) and so
\[
  f(m) \leq 3 + \sum_{i \in J} f(m_i) \leq 3 + \sum_{i \in J} \frac{6m_i}{k} - 3\|J\| \leq \frac{6m}{k} - 3
\]
Quad-BSP Tree Analysis (concluded)

Lemma 12.7 A leaf region intersects at most $k + 4\lambda$ objects.

Proof

- Squares are covered by Lemma 12.6.
- A quadtree split creates square regions.
- A shrinking step creates two squares and two rectangles.
- The exterior of one square $\sigma'$ contains at most $k$ guards.
- Each rectangle $\sigma''$ is inside a square in the exterior of $\sigma'$. 
Density Estimation

The density $\lambda$ is unknown and is hard to compute.
We estimate $\lambda$ within a factor of two.

1. Set $\lambda = 2$.
2. Construct the quad-BSP tree with $k = \lambda$.
3. If no leaf contains more than $5\lambda$ fragments, return the tree.
4. Double $\lambda$ and go to step 2.
We can compute the BSP of $n$ line segments with the density sensitive algorithm using the auto-partition algorithm for phase 3.

**Theorem 12.8** The BSP has size $O(n \log \lambda)$.

*Proof*

- Phase 1 gives a quad-BSP tree with $O(n/\lambda)$ leaves.
- Phase 2 inserts at most $5\lambda$ subsegments into each leaf.
- Phase 3 expands each leaf into a subtree of size $O(\lambda \log \lambda)$.

**Theorem 12.9** An analogous algorithm constructs a BSP of size $O(n\lambda)$ for $n$ 3D triangles.

These results are optimal.