Binary Space Partitions

Latest Developments

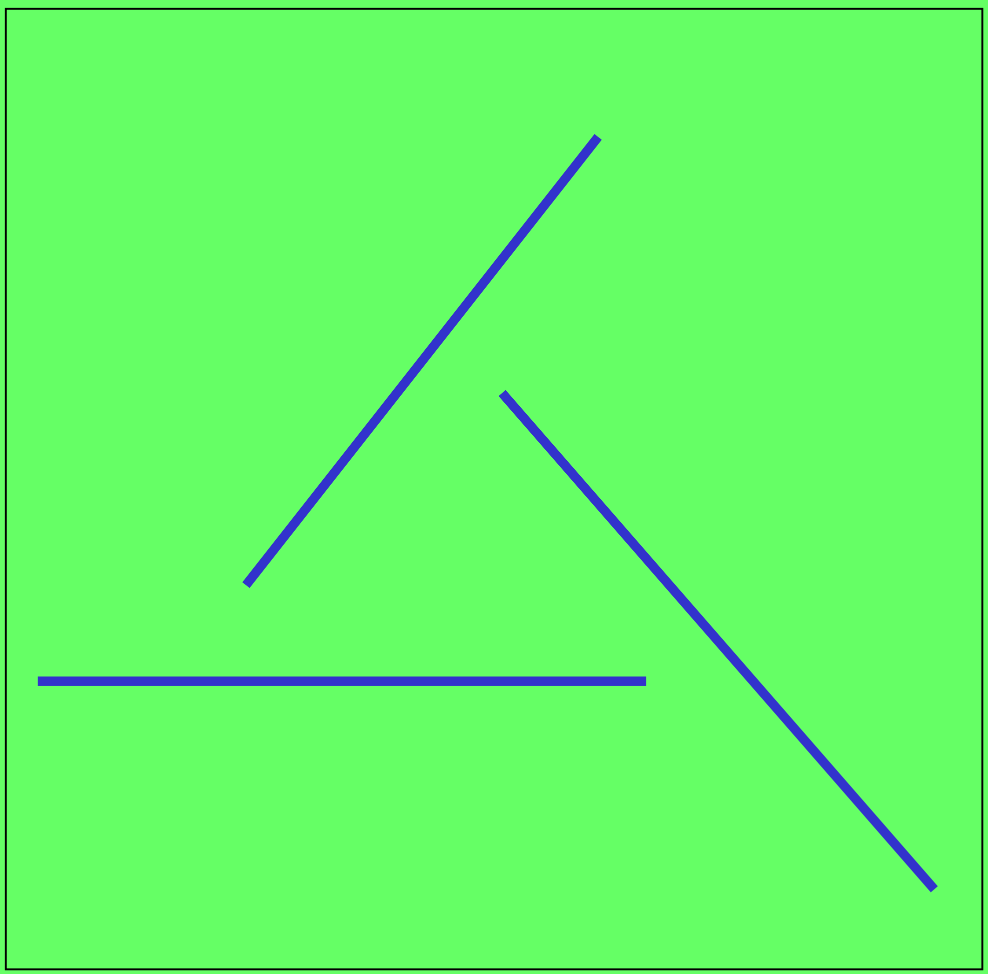
Csaba D. Tóth

toth@cs.ucsb.edu

UCSB

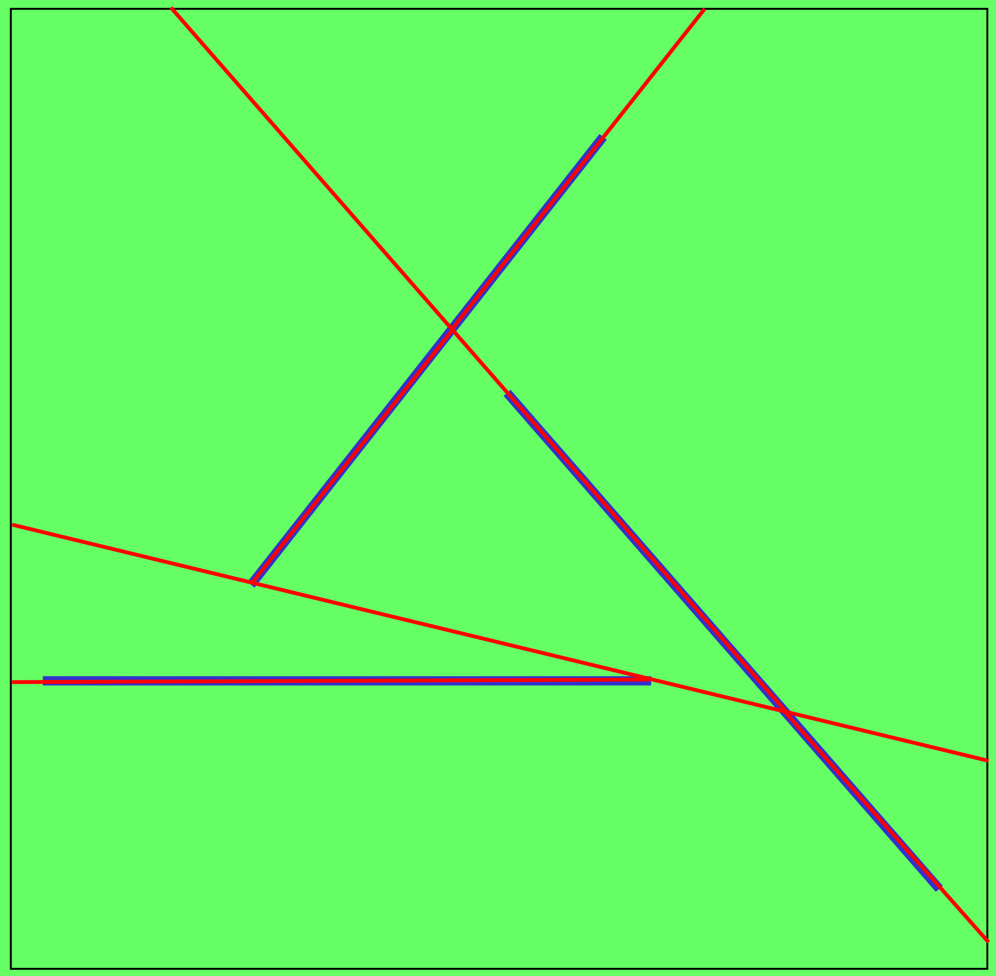
Binary Space Partition

for a set of *(d-1)*-dimensional objects in $\mathbf{R}^d$



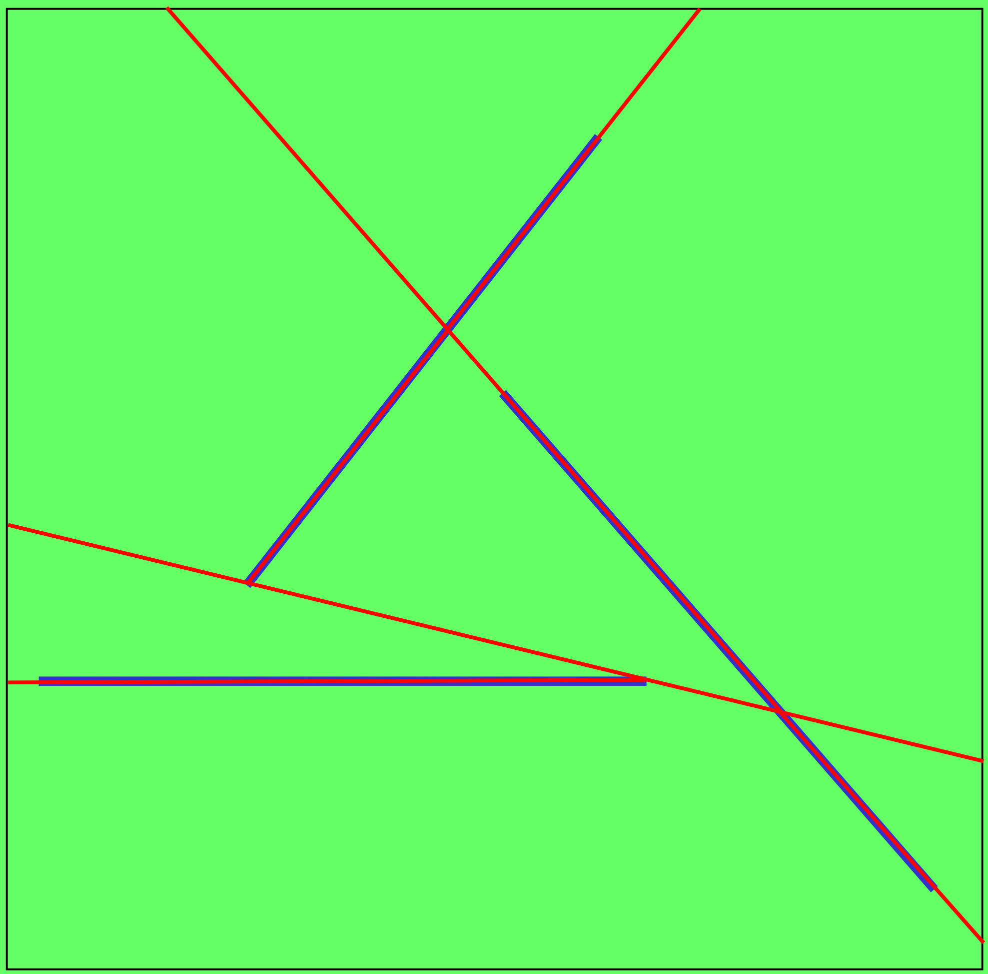
Binary Space Partition

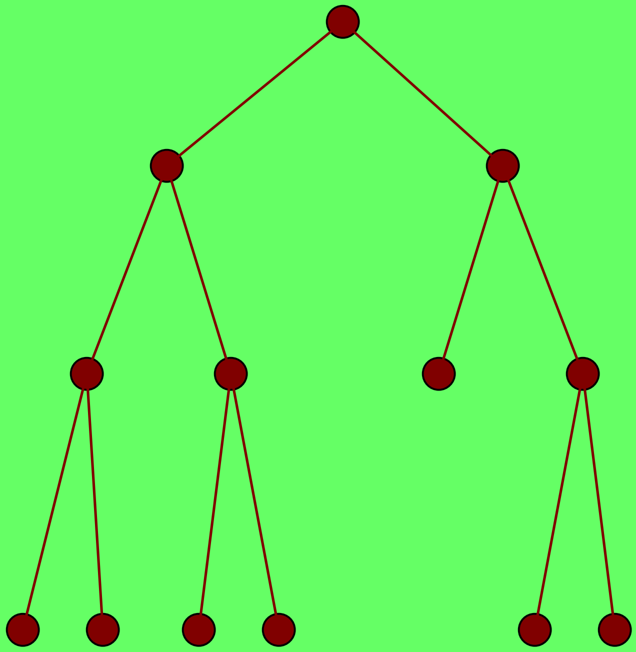
for a set of *(d-1)*-dimensional objects in $\mathbf{R}^d$



Binary Space Partition

for a set of *(d-1)*-dimensional objects in $\mathbf{R}^d$

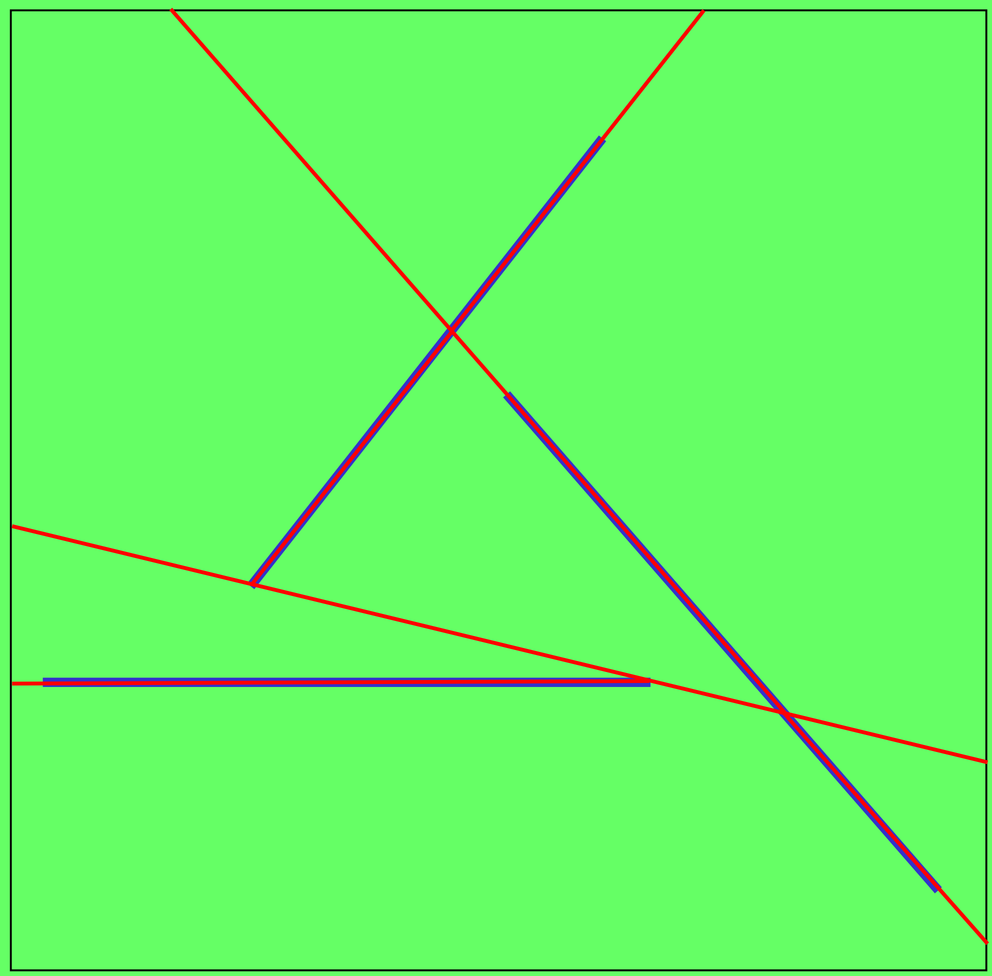


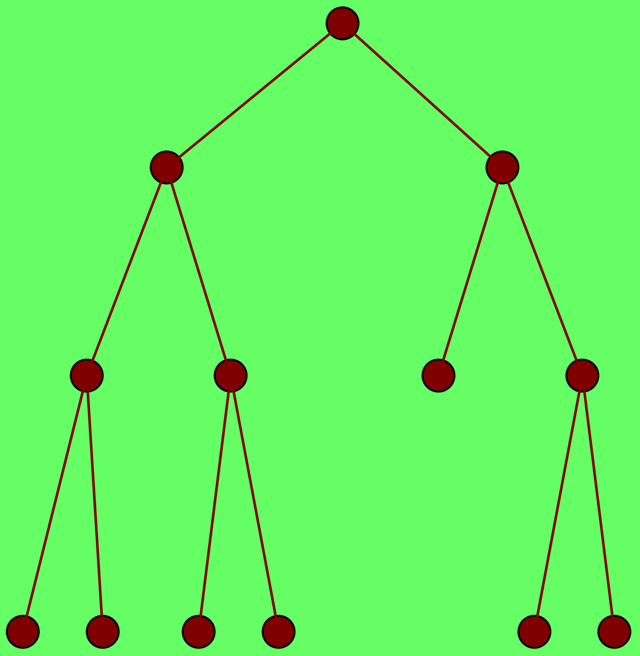


Size of the BSP:
The number of nodes in the BSP tree.

Binary Space Partition

for a set of *(d-1)*-dimensional objects in $\mathbf{R}^d$

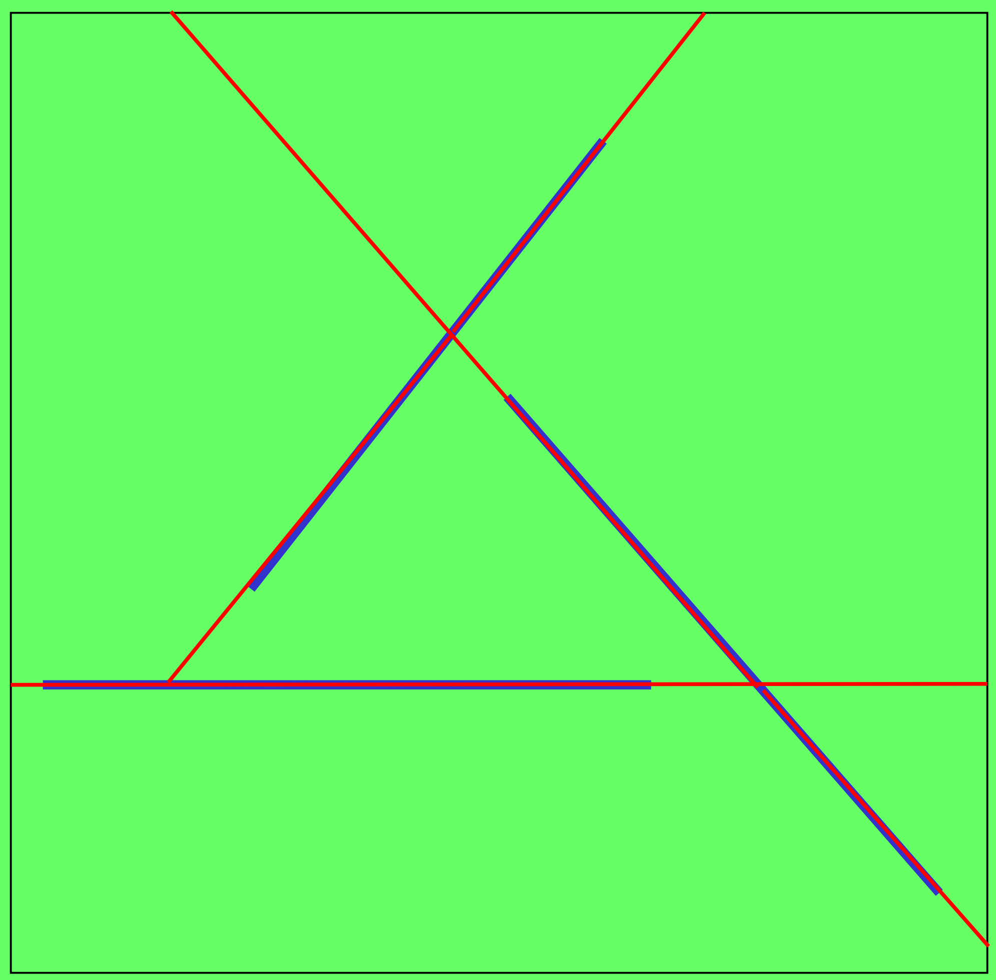


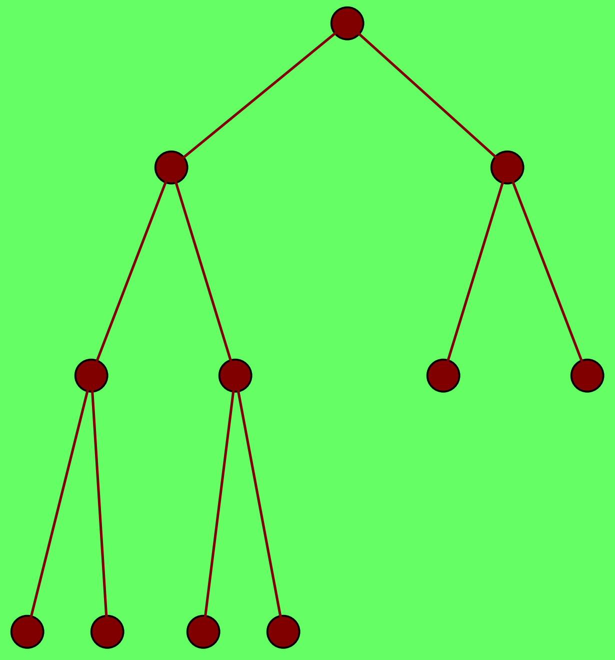


Size of the BSP ≈
≈ The number of fragments of input objects.

Auto-Partition

for a set of *(d-1)*-dimensional objects in $\mathbf{R}^d$

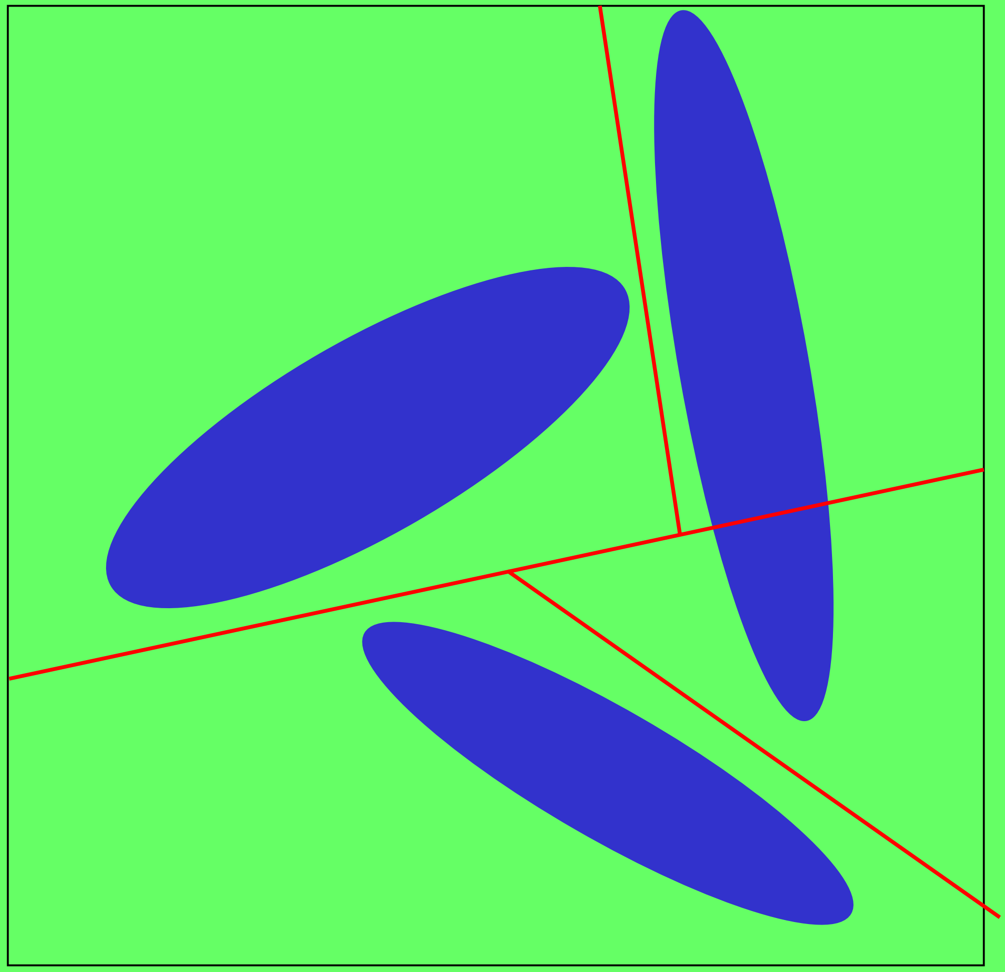


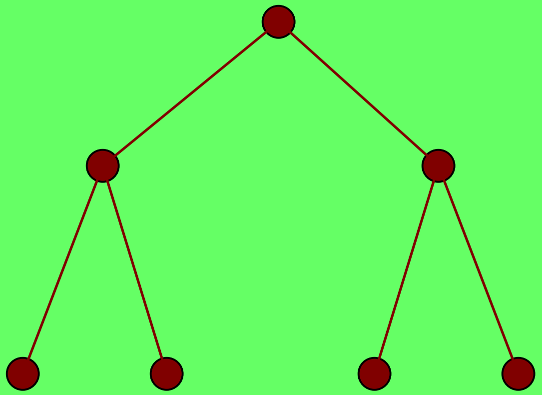


Cutting hyper-planes lie along input objects.

Binary Space Partition

for a set of *full*-dimensional objects in $\mathbf{R}^d$

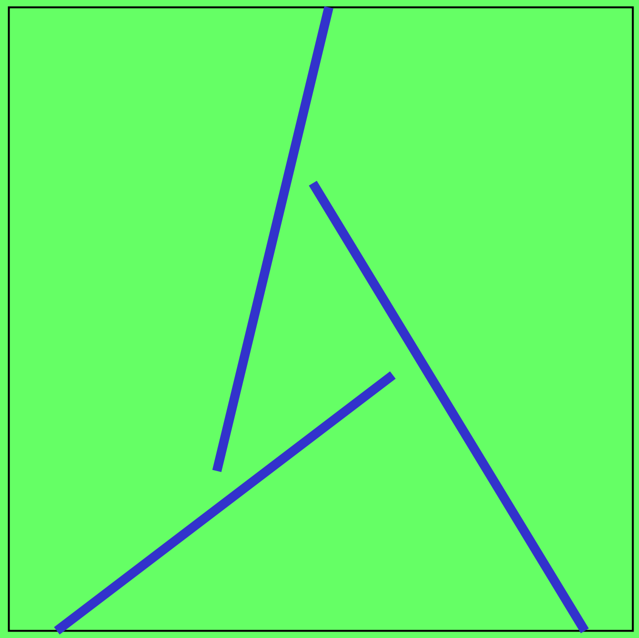




The interior of every cell intersects at most one input object.

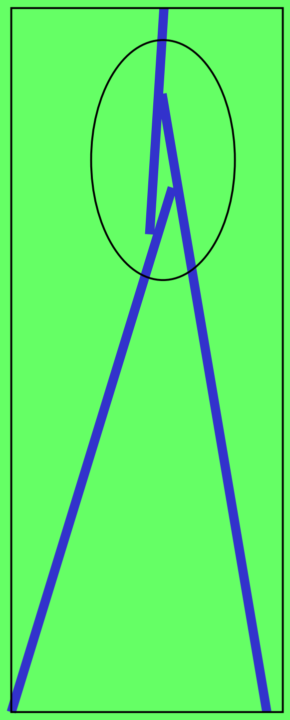
Binary Space Partition

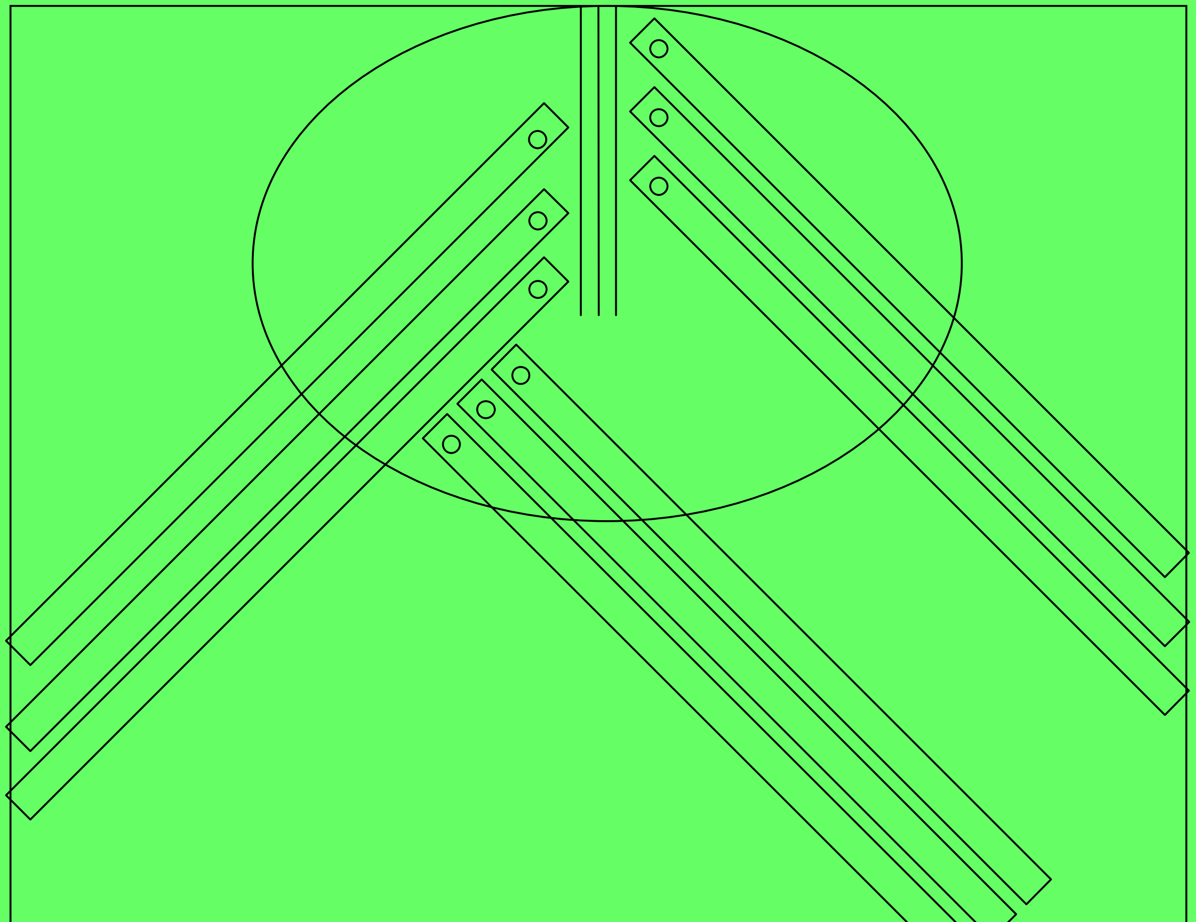
- Paterson and Yao (1989): The size of the smallest BSP for n disjoint segments in $\mathbf{R}^2$ is $O(n \log n)$.
- T. (2001): Best known lower bound $\Omega(n \log n / \operatorname{loglog} n)$.



Binary Space Partition

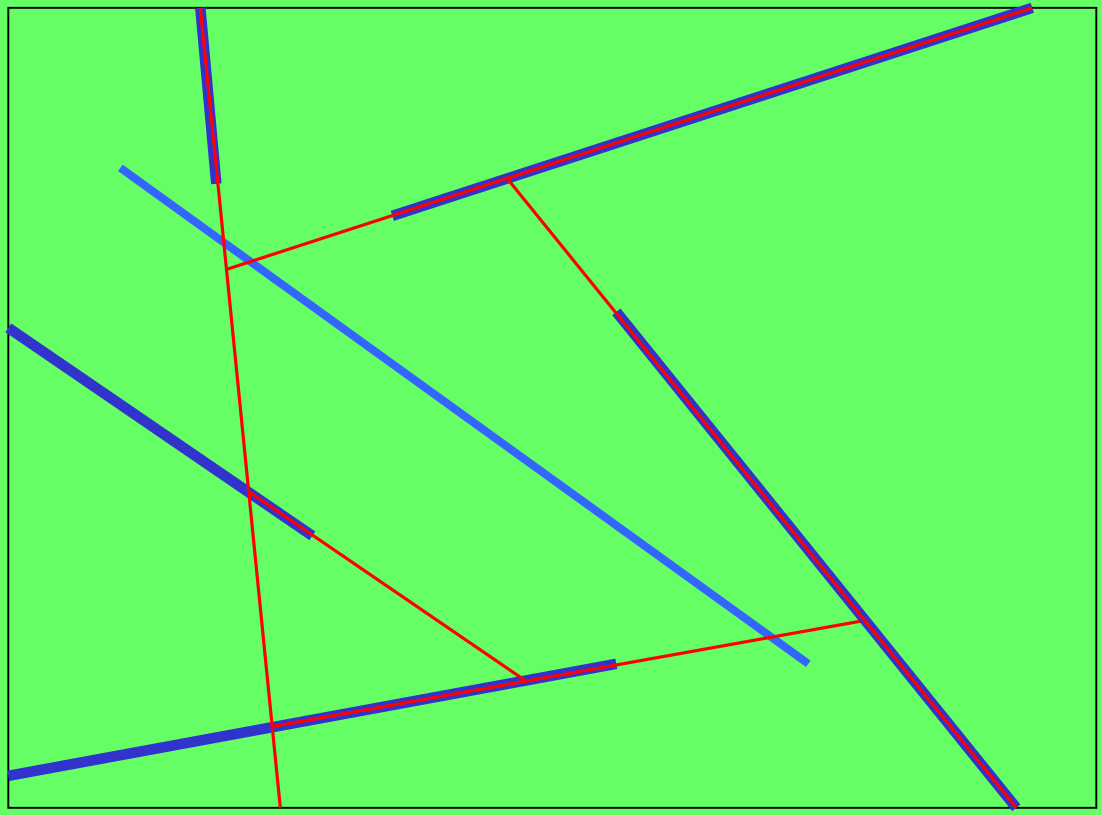
- Paterson and Yao (1989): The size of the smallest BSP for n disjoint segments in $\mathbf{R}^2$ is $O(n \log n)$.
- T. (2001): Best known lower bound $\Omega(n \log n / \log\log n)$.





Binary Space Partition

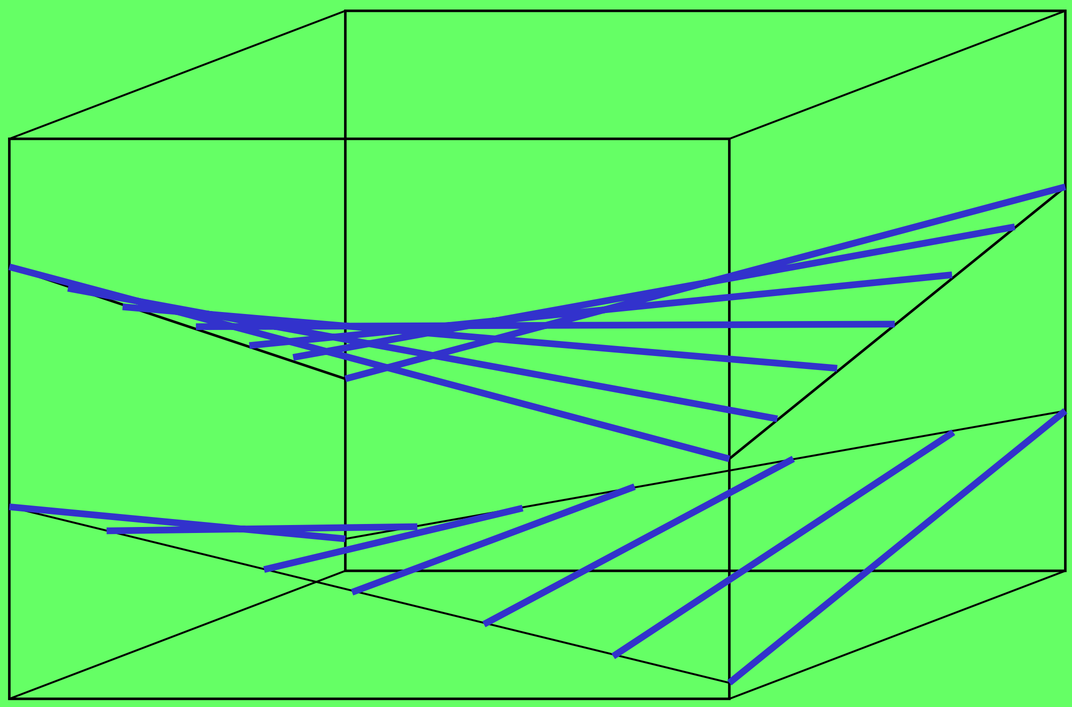
- Paterson, Yao (1990): Axis-parallel segments in $\mathbf{R}^2$: $\Theta(n)$.
- T. (2002): Line segments with k different orientations in $\mathbf{R}^2$: $O(n \log (k+1))$.



Binary Space Partition

—3-dimensions—

- Paterson and Yao (1989): Segments in $\mathbf{R}^d$, $d>2$: $\Theta(n^2)$.



No super-quadratic lower bound is known for the size of any BSP.

Binary Space Partition

—Orthogonality helps—

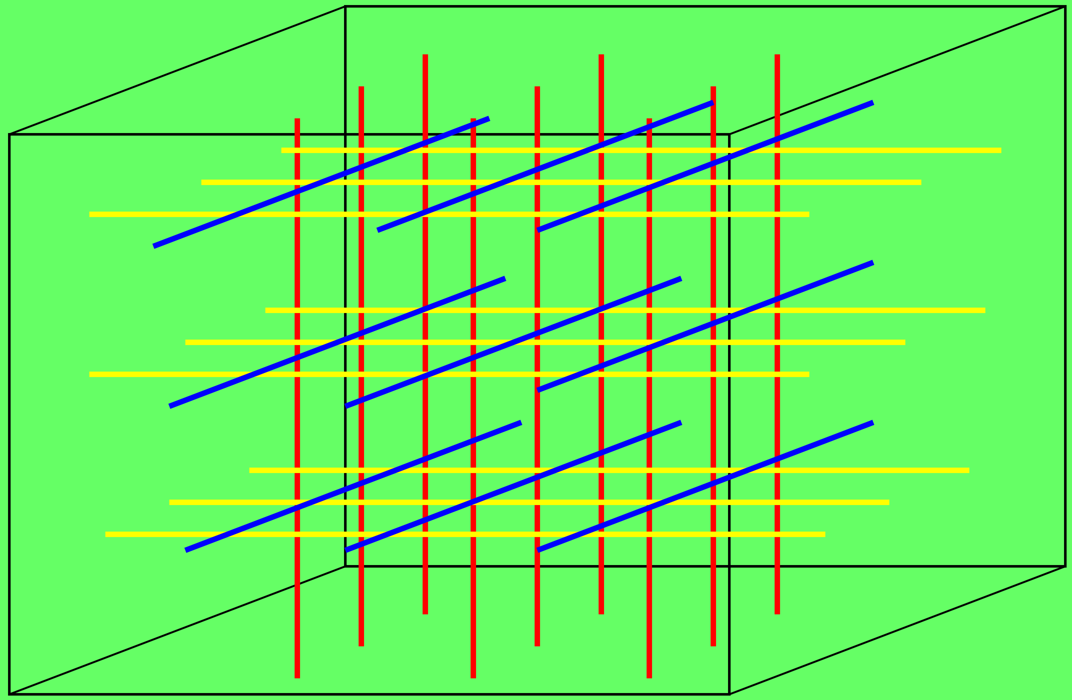
- Paterson, Yao (1990): Axis-parallel segments in $\mathbf{R}^2$: $\Theta(n)$.
- D'Amore and Franciosa (1992) & Dumitrescu et al. (2001): Axis-parallel segments in $\mathbf{R}^2$: $[2n-o(n), 2n-1]$
- Arya (2002): size-height tradeoff – the size of a BSP tree of height h is $\Omega(n \log n / \log h)$.

- Dumitrescu, Mitchell, Sharir (2001) & Berman, DasGupta, Muthukrishnan (2001): Axis-parallel rectangles in $\mathbf{R}^2$: $[7n/3-o(n), 3n-o(n)]$.
- Axis-parallel space filling rectangles: $[2n/3-o(n), 2n]$.

Binary Space Partition

—Orthogonality in higher dimensions—

- P&Y(1990): Axis-parallel segments in $\mathbf{R}^d$, $d>2$: $\Theta(n^{d/(d-1)})$.

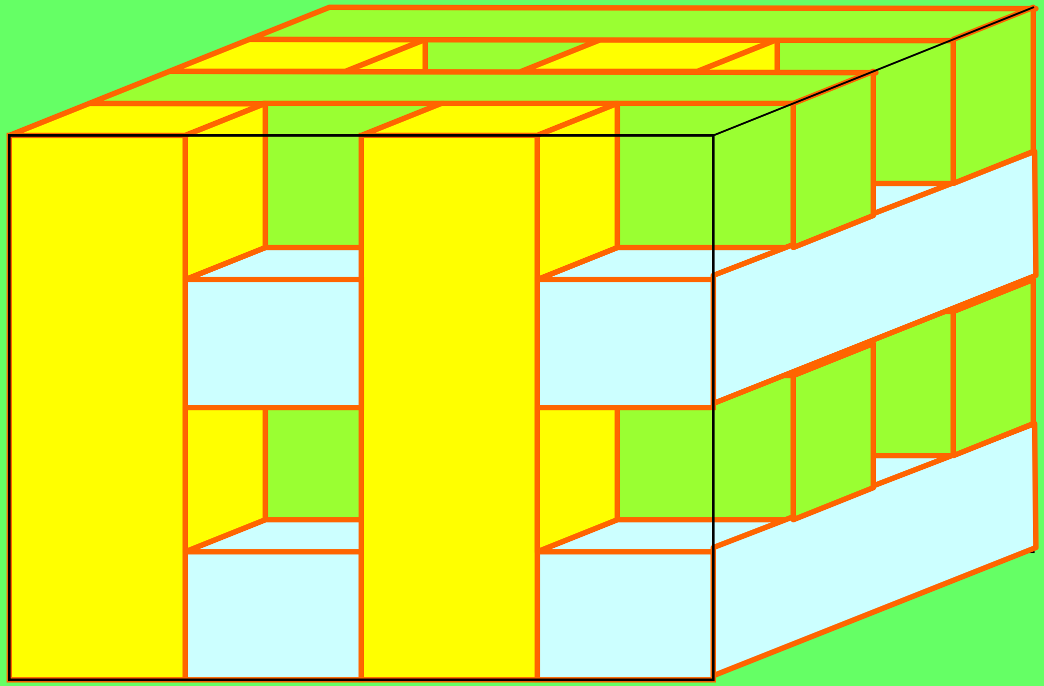


- Dumitrescu, Mitchell, Sharir (2001): Axis-parallel k-dimensional rectangles in $\mathbf{R}^d$: $\Theta(n^{d/(d-k)})$ if $k < d/2$, but $O(n^{d/(d-k)})$ holds for every $k \leq d\text{-}1$.

Binary Space Partition

— Space filling helps—

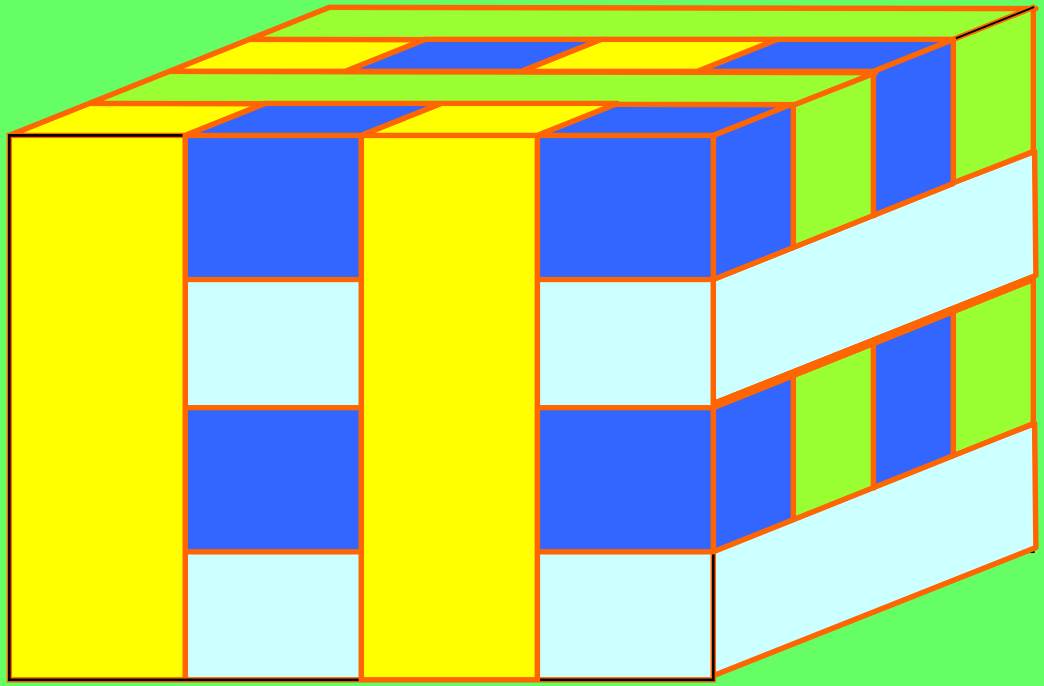
Space filling increases the complexity of the input by the minimum convex partition of its complement.



Binary Space Partition

— Space filling helps—

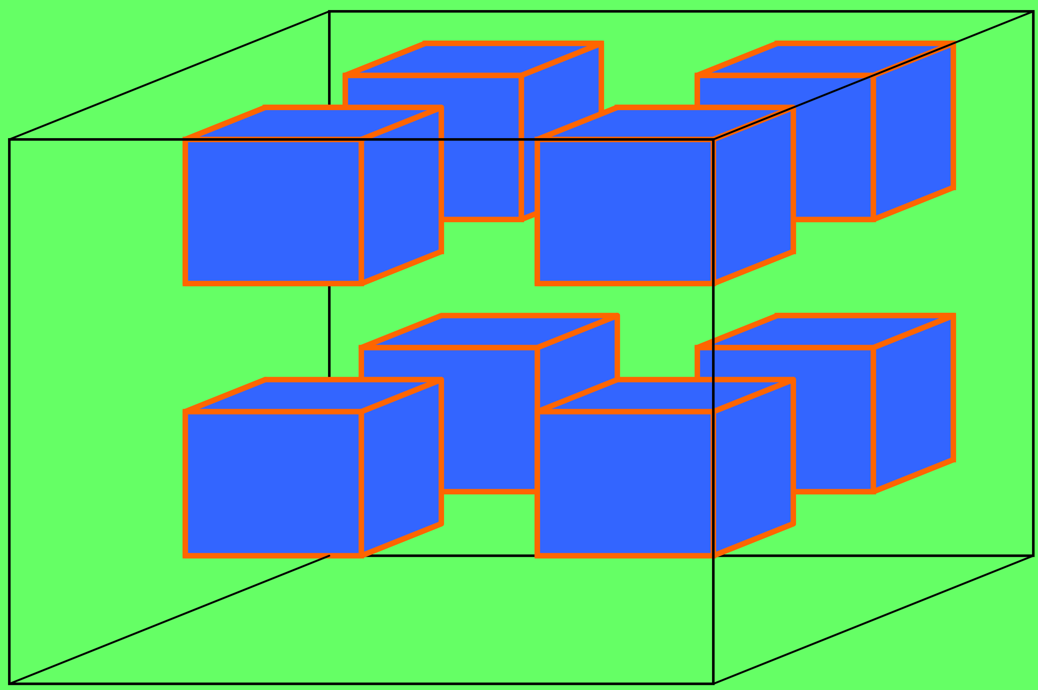
Space filling increases the complexity of the input by the minimum convex partition of its complement.



Binary Space Partition

— Space filling helps—

Space filling increases the complexity of the input by the minimum convex partition of its complement.



The complement of $3 \cdot k^2$ prisms consists of k^3 small cubes.

Binary Space Partition

— Space filling helps—

Hershberger, Suri, and T. (2003):
Space-filling axis-parallel boxes in $\mathbf{R}^3$: $\Theta(n^{4/3})$.

Two phase algorithm:

- BSP for the *vertices* of all input boxes (by “round-robin”),
- *linear* BSP for objects whose vertices are on cell boundary.

Hershberger, Suri, and T. (2003):
Space-filling axis-parallel boxes in $\mathbf{R}^d$: $O(n^{(d+1)/3})$ and $\Omega(n^{\beta(d)})$ where $\beta(d) \to (1+\sqrt{5})/2$.

Binary Space Partition

—Fatness helps—

- De Berg, de Groove, and Overmars (1997): Ratio of longest and shortest segments is bounded by a constant in $\mathbf{R}^2$: $\Theta(n)$.
- De Berg (2000): Full dimensional fat objects in $\mathbf{R}^d$: $\Theta(n)$.

Two phase algorithm:
- BSP for *bounding box vertices* of all fat objects,
- BSP for *constant number* of objects in every cell.

Generalizes to *uncluttered scenes* in $\mathbf{R}^d$: $\Theta(n)$.

- Agarwal, Grove, Murali, and Vitter (2000): Fat axis-parallel rectangles in $\mathbf{R}^3$: $n2^{O(\sqrt{\log n})}$.

Binary Space Partition

—Fatness helps—

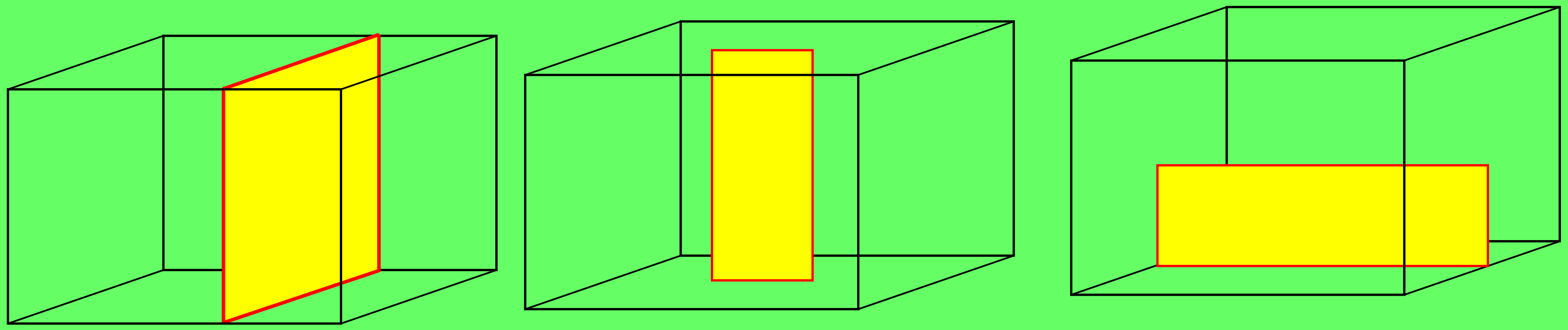
T. (2003): Axis-parallel fat rectangles in $\mathbf{R}^3$: $O(n \log^8 n)$ *and* $\Omega(n \log n)$ (for orthogonal BSP).

T. (2003): Axis-parallel fat (d-1)-dimensional rectangles in $\mathbf{R}^d$: $O(n$ polylog(d) $n)$.

Fat Rectangles

Given a box C in $\mathbf{R}^3$, we say that an 2-dimensional rectangle r is

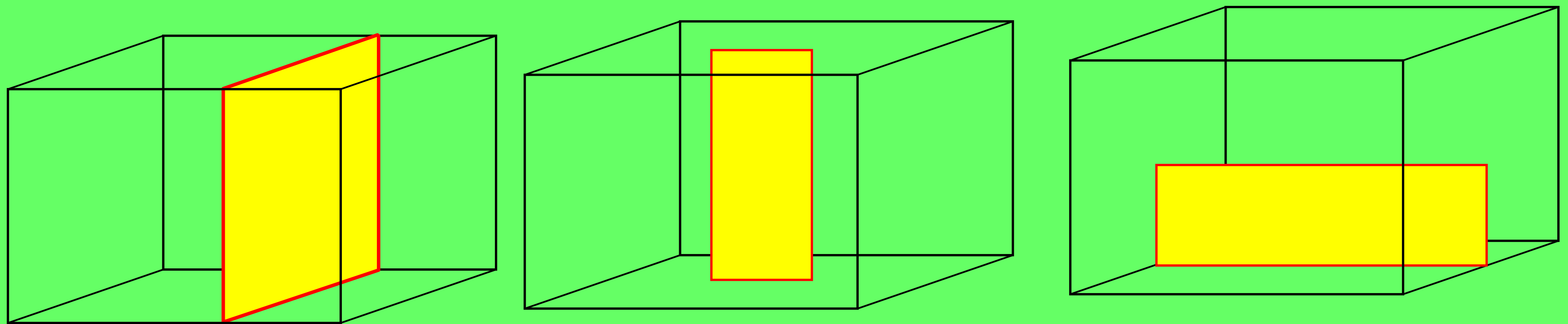
- **Long**, if no vertices of r are in int(C) [an extent contains extent of C].
- **Free-cut**, if two extents of r contains the corresponding extents of C.
- **Bridge**, if *one* extent of r contains that of C, and the other extent is in the interior of the corresponding extent of C.
- **Shelf**, if *one* extent of r contains that of C, and the other extent contains an endpoint of the corresponding extent of C.



Fat Rectangles

The rectangle $r \cap C$ is not necessarily fat, but

- If *r* is a free-cut, then we can partition *C* along *r*, without cutting any other objects. $\Rightarrow$ We can assume that there's no free-cut.
- If *r* is a bridge, then the extent of *r* within *C* is at most α-times shorter than the shortest of the other extent (*semi-fatness*).



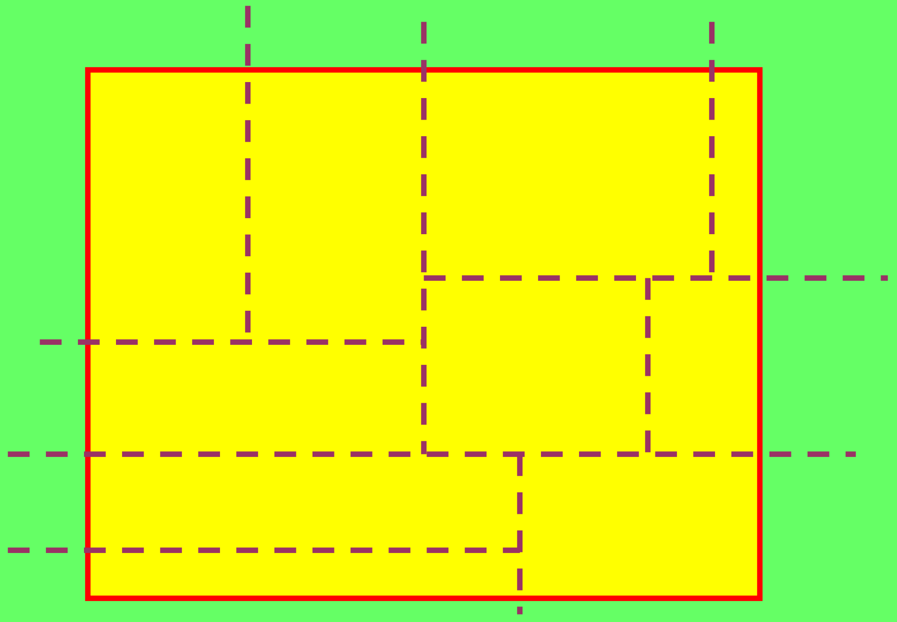
Clipped Segments

It is not difficult to find an $O(n)$ size BSP for n long rectangles.

This BSP can fracture the other fat rectangles into many pieces.

Lemma:

There is a BSP for n long fat rectangles such that every clipped axis-parallel segment is cut into $O(\log^3 n)$ pieces.



If every clipped segment is cut into $O(\log^{\alpha} n)$ pieces, then every rectangle is cut into $O(\log^{2\alpha} n)$ pieces.

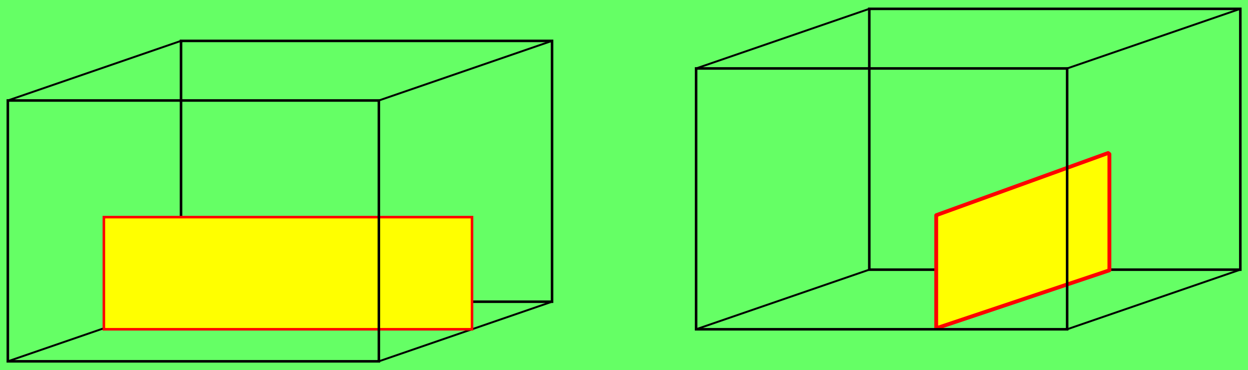
BSP for rectangles in R^3

Algorithm

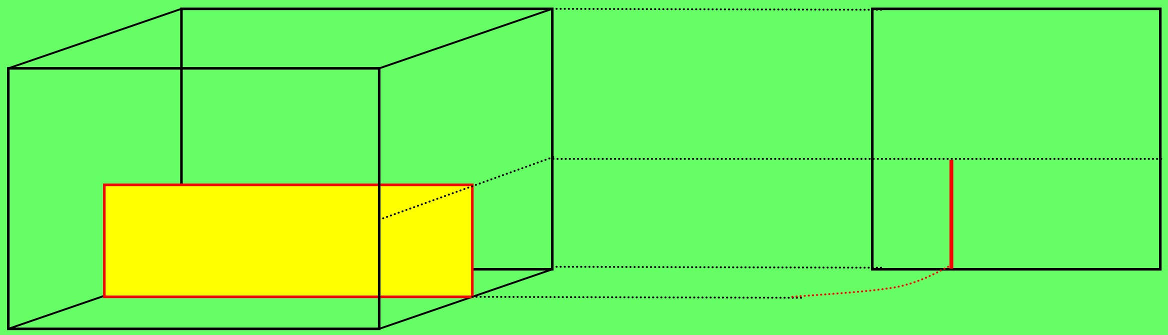
- Divide the bounding box C into $8=2^3$ subproblems along medians of the vertices of the rectangles,
- Overlay a BSP for **long** rectangles, while cutting every clipped segment into $O(\log^3 n)$ pieces,
- Process the subproblems recursively.

BSP for Shelves

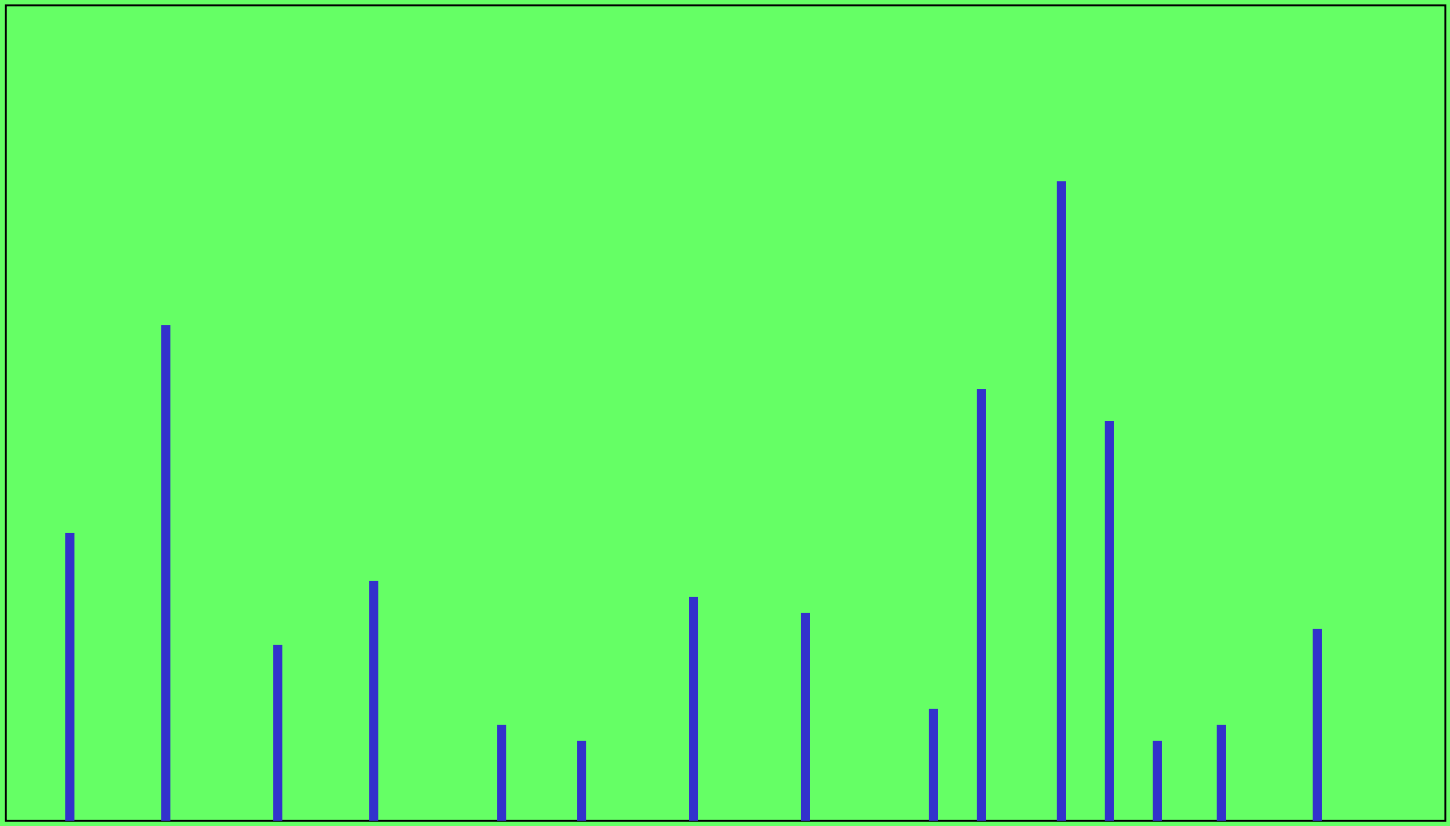
All shelves along one side of *C* are parallel.



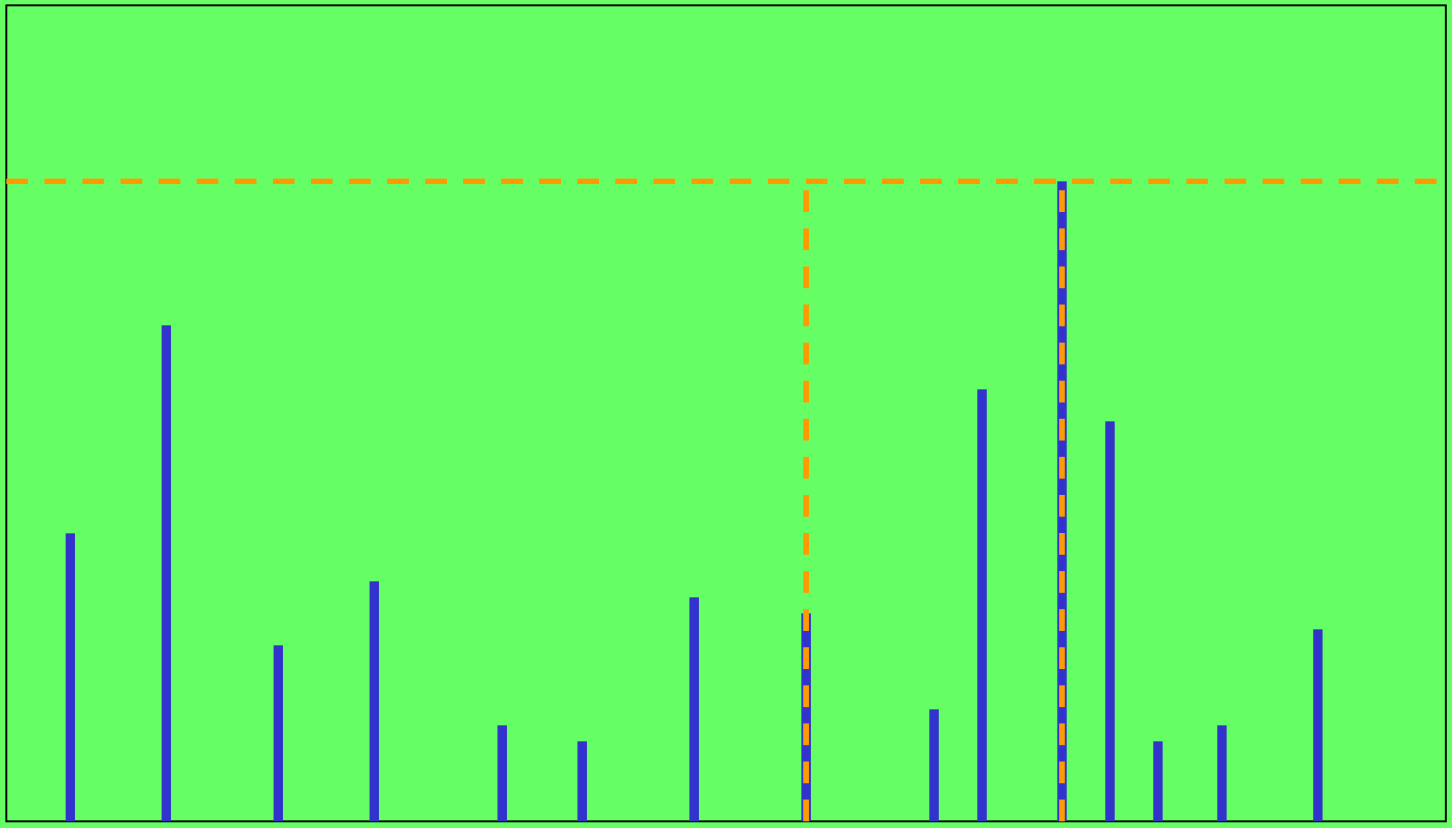
All shelves along one side of *C* can be represented in $\mathbf{R}^2$.



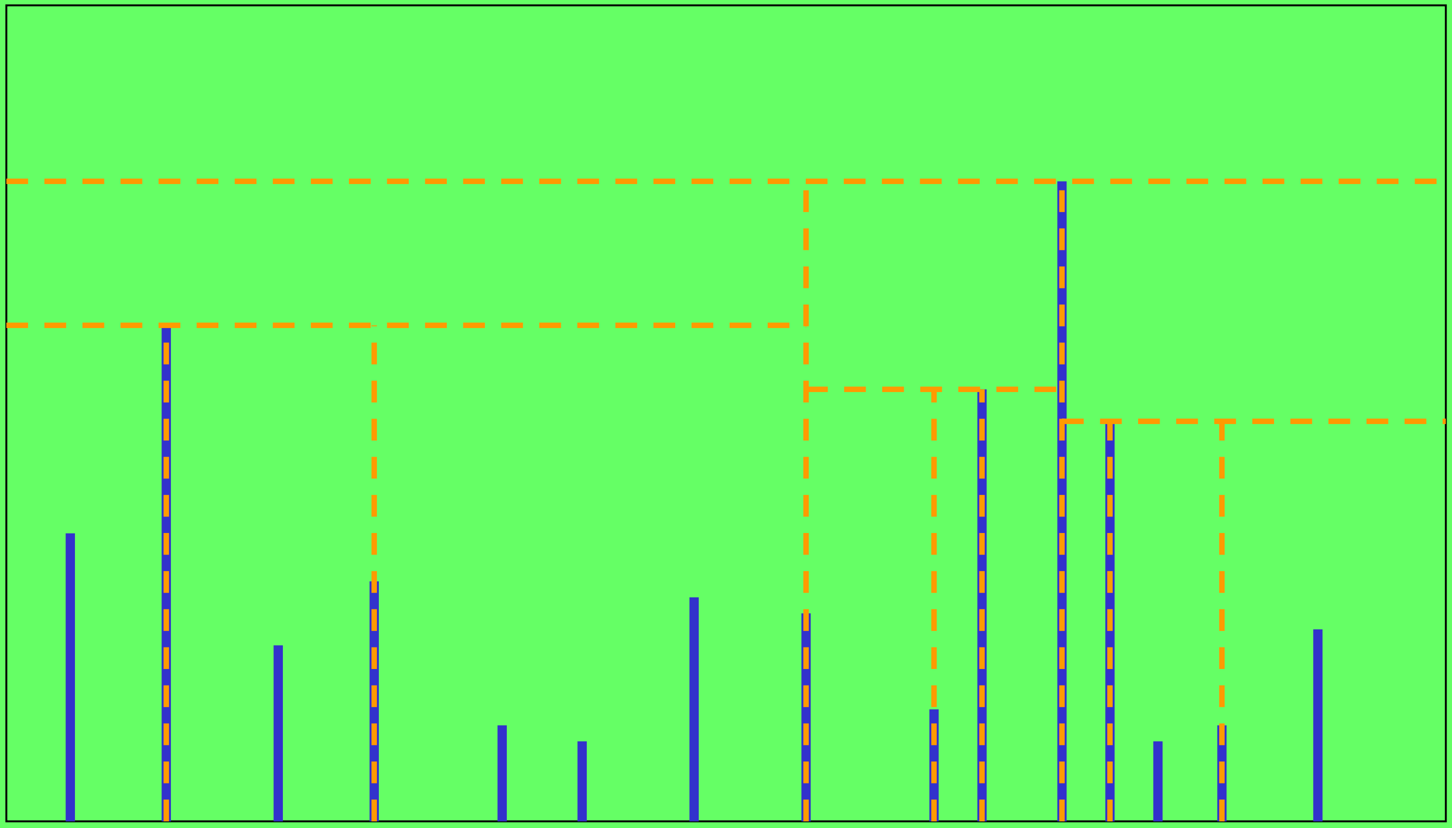
BSP for Shelves



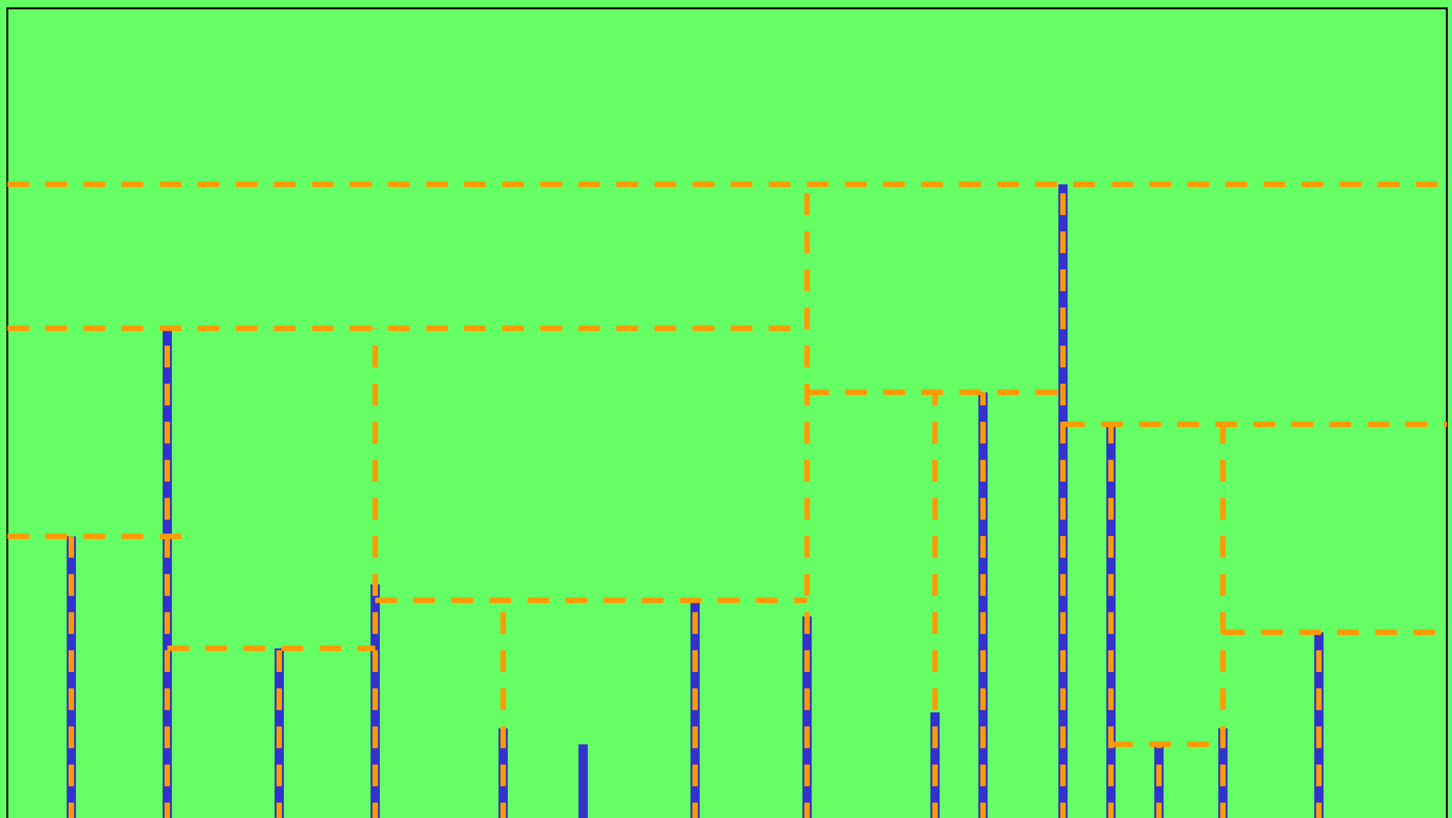
BSP for Shelves



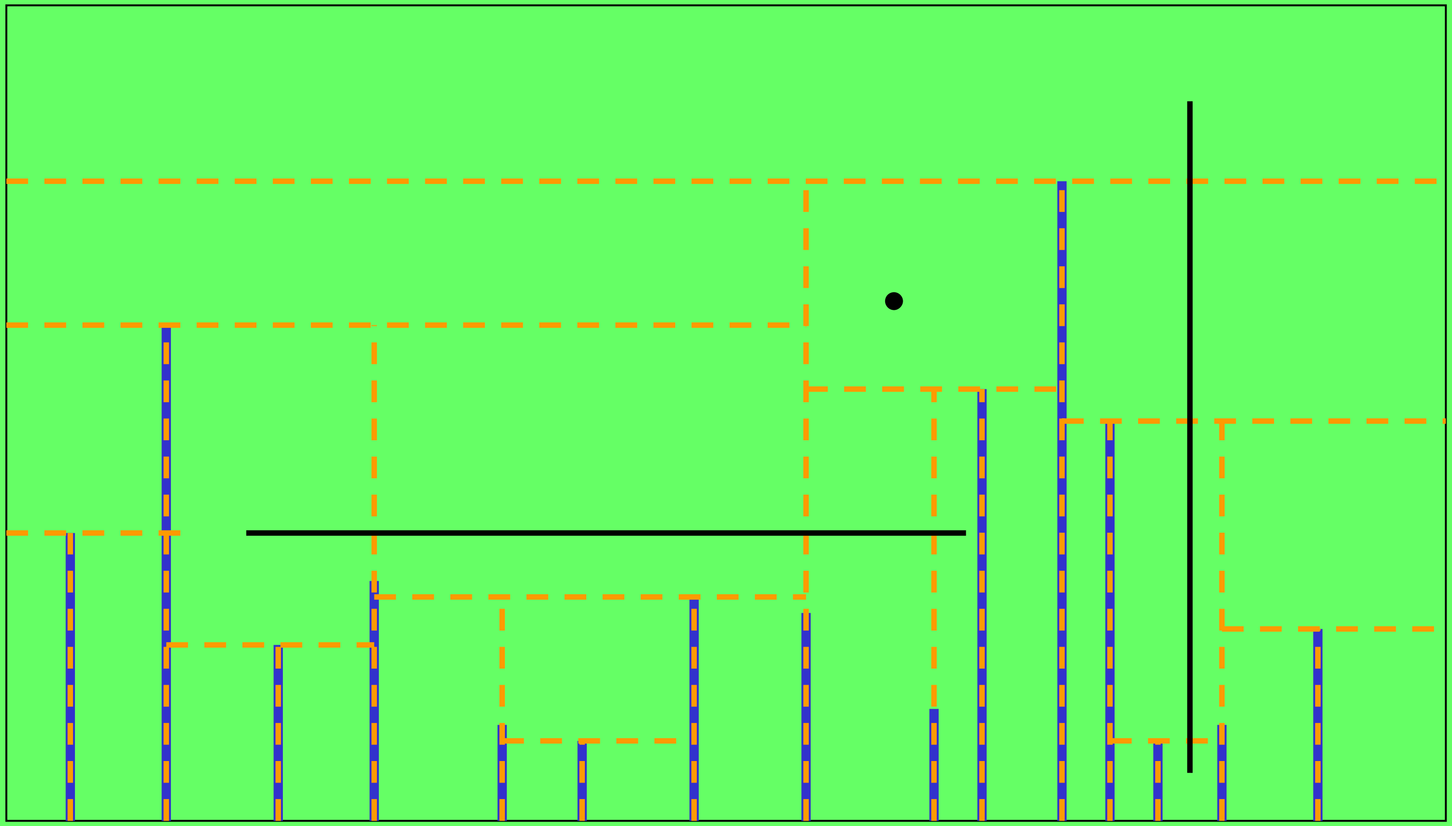
BSP for Shelves



BSP for Shelves



BSP for Shelves

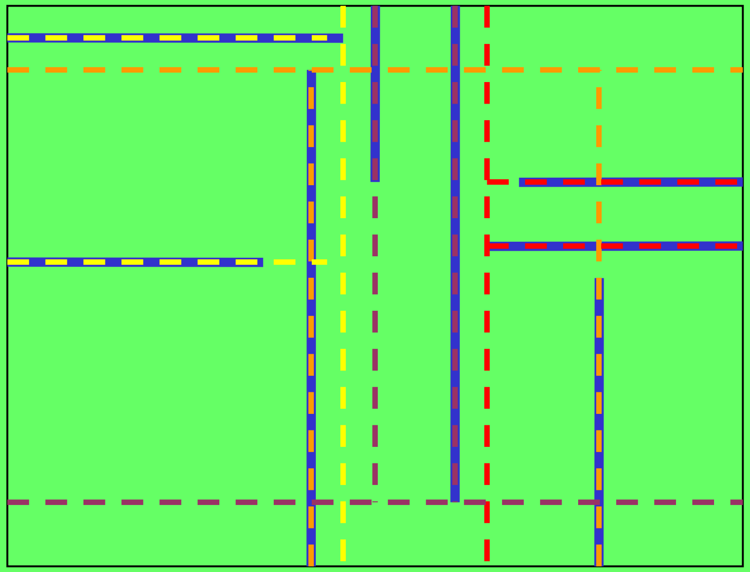


BSP algorithm ends in log n rounds.

Every axis-par. segment clipped to a rectangle is cut into $O(\log n)$ pieces.

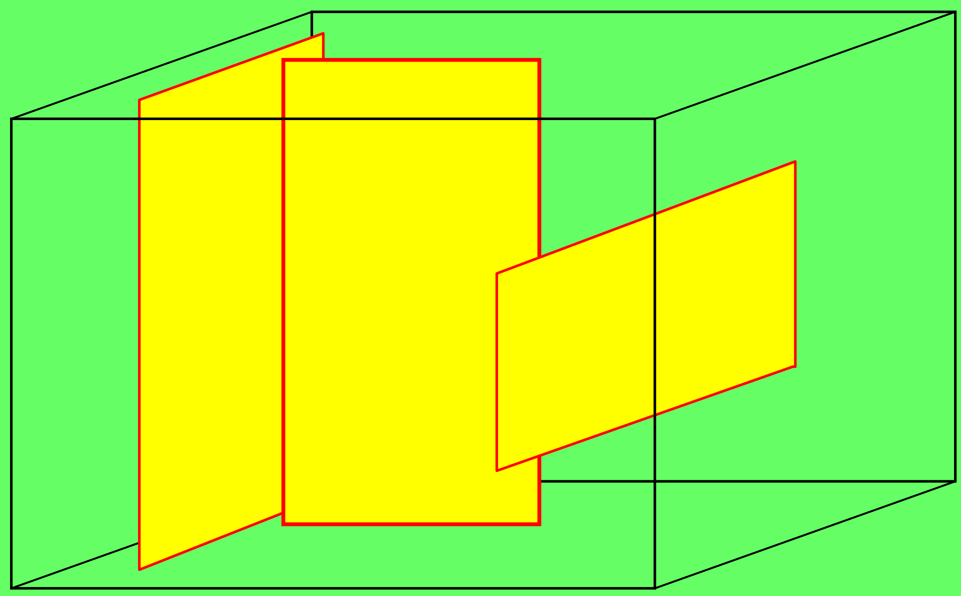
Overlay of BSPs

Apply the BSP for shelves independently on every side of a box C.

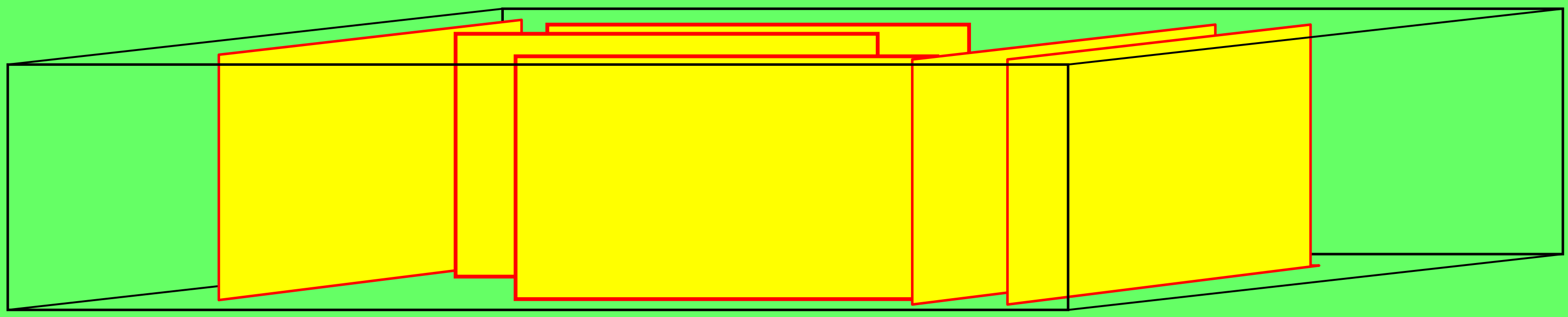


If we apply k BSPs on the same domain where each BSP cuts every axis-parallel clipped segment λ times, then the **overlay** is a BSP that cuts every axis-parallel clipped segment at most $k\lambda$ times.

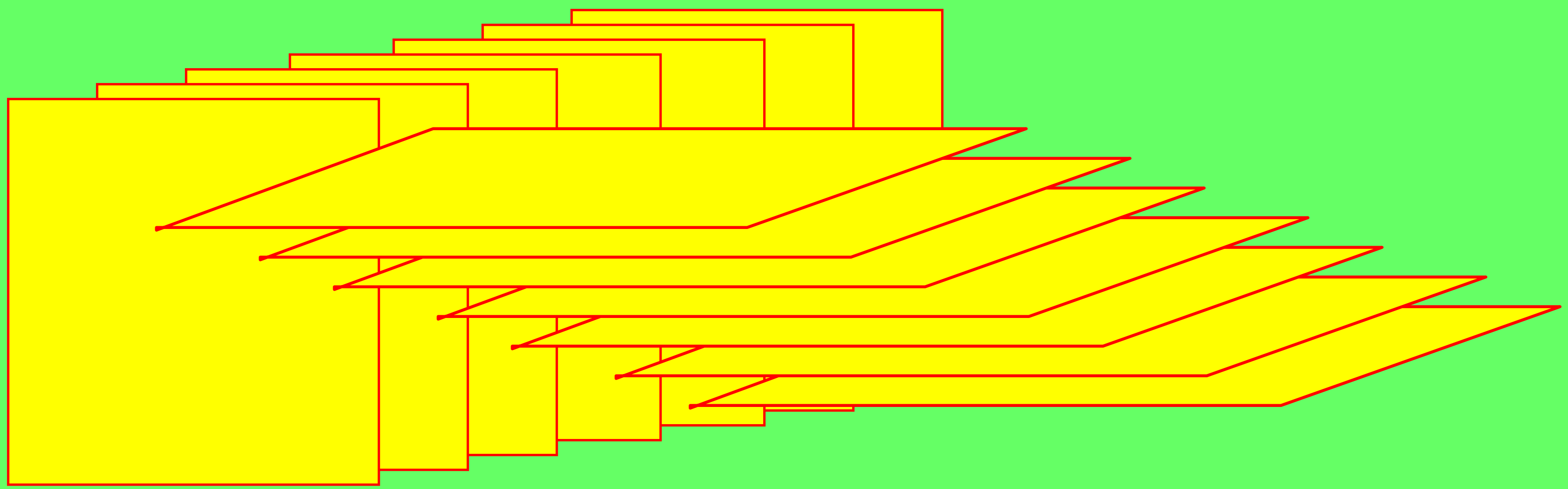
BSP for long rectangles in R^3



Bridges with a common direction behave like axis-par. segments in $\mathbf{R}^2$.



Lower Bound Construction



BSPs for lower-dim fat objects

$(d\text{-}1)$-dimensional fat axis-parallel hyper-rectangles in $\mathbf{R}^d$ have the same BSP complexity as axis-parallel line segments in the plane — apart from a poly-logarithmic factor.

Is it true for every k, $1 < k < d\text{-}1$, that n disjoint k-dimensional fat axis-parallel hyper-rectangles in $\mathbf{R}^d$ have the same BSP size (apart from a poly-logarithmic factor) as n axis-parallel line segments in $\mathbf{R}^{d-k+1}$, that is, $\Theta(n^{(d-k+1)/(d-k)}$ polylog $n)$.