OI-OPPA. European Social Fund
Prague & EU: We invest in your future.

Partition Trees

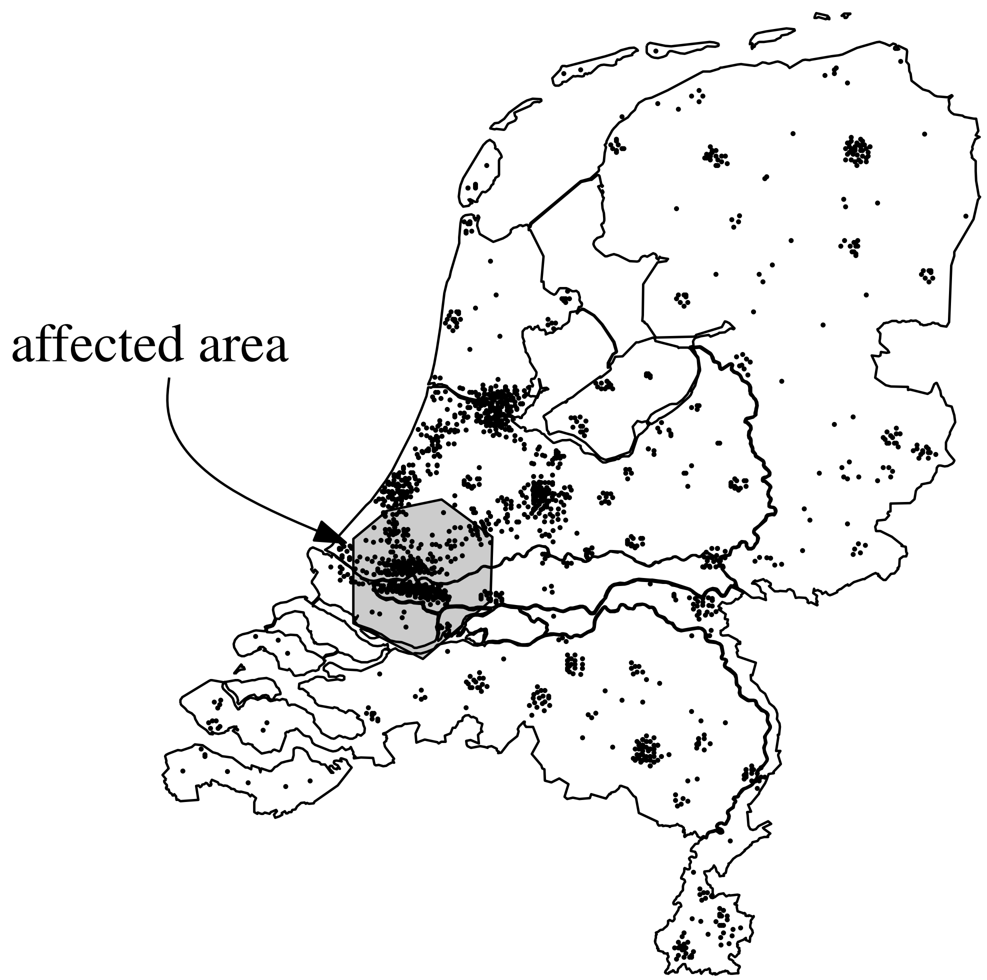
J Pritts 29.11.2012

ae4m39vg

Overview

- Used to quickly $O(\sqrt{n})$ count points in an arbitrary region
- A query range is any subset of the plane that can be approximated by a simple polygon
- Part of the family of geometric range searching algorithms, specifically, it is called a simplex range search (you will see why)

Application



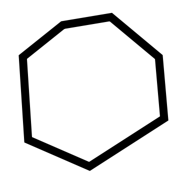
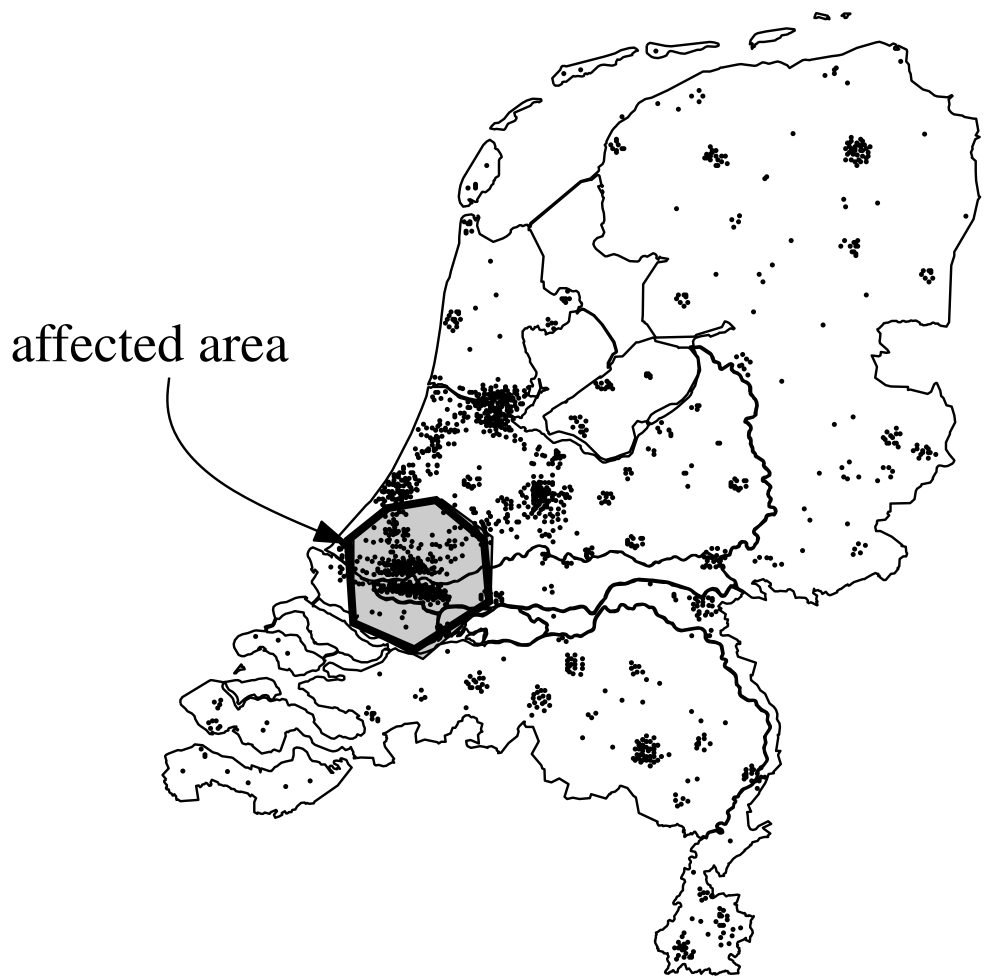
Query range defined by wind patterns results in a  non-orthogonal query

Figure 16.1
Population density of the Netherlands

Application



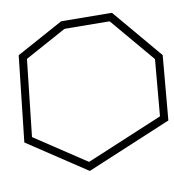
Query range defined by wind patterns results in a  non-orthogonal query

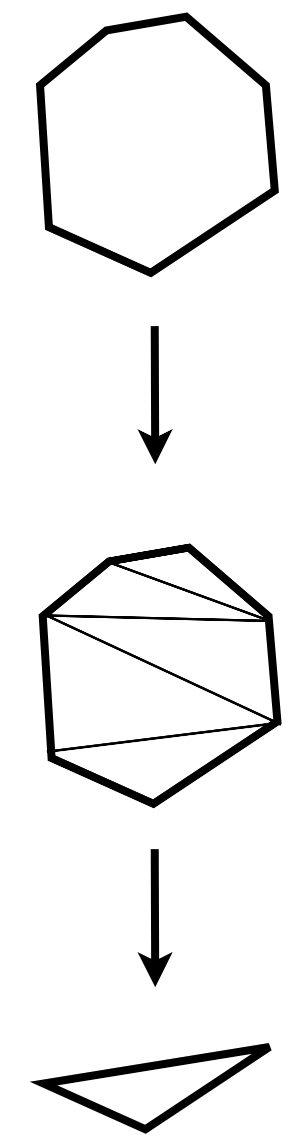
Figure 16.1
Population density of the Netherlands

Simplifying the query

- Query regions can be arbitrarily large and complex

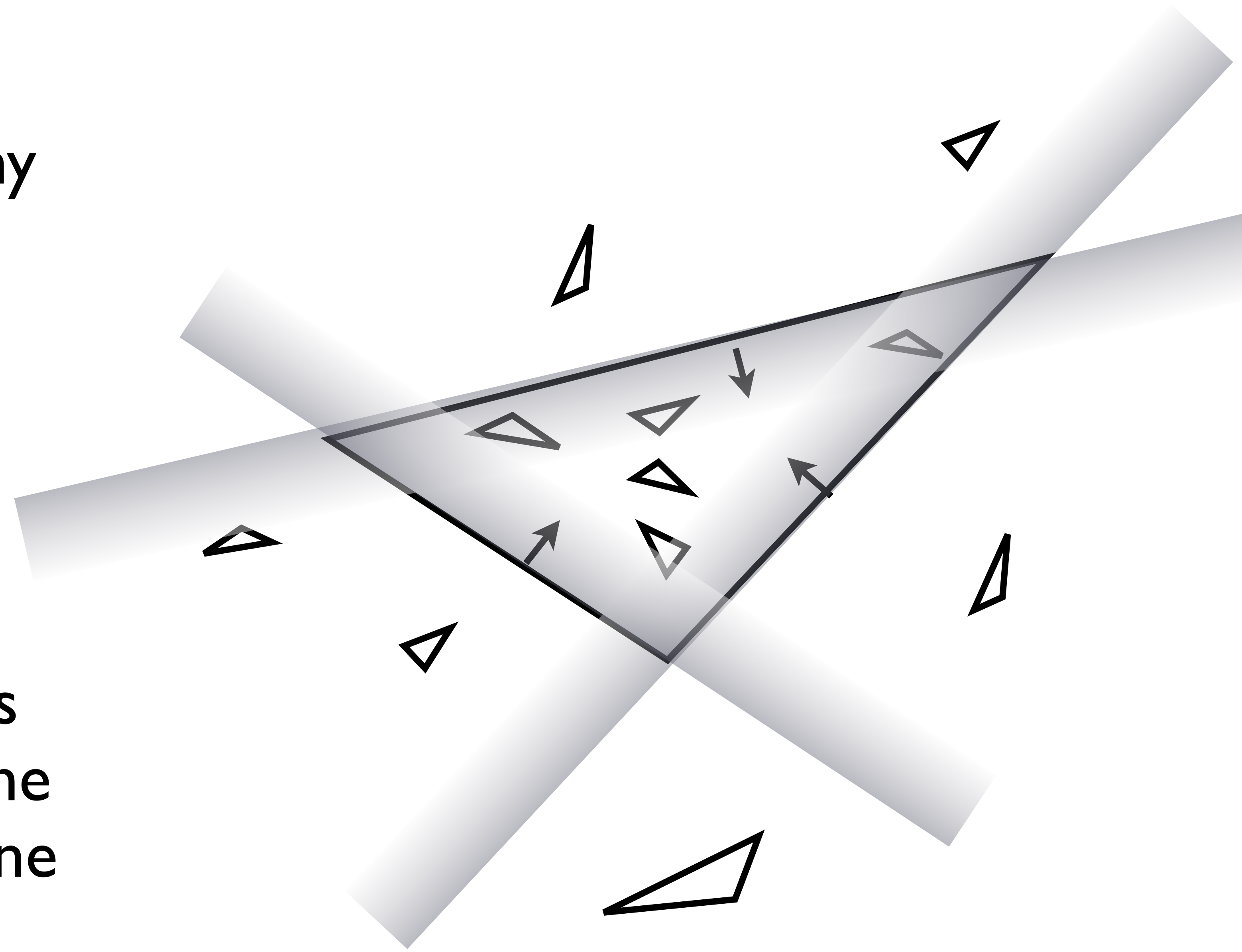
- Replace representation of polygon by a union of disjoint closed simplexes (triangles in 2D)

- Range search on query region is sum of range searches on triangles



Counting triangles

- Need to find how many triangles are in the interior of a simplex
- Create three sets corresponding to the three half planes, each containing the triangles which are subsets of the corresponding half-plane

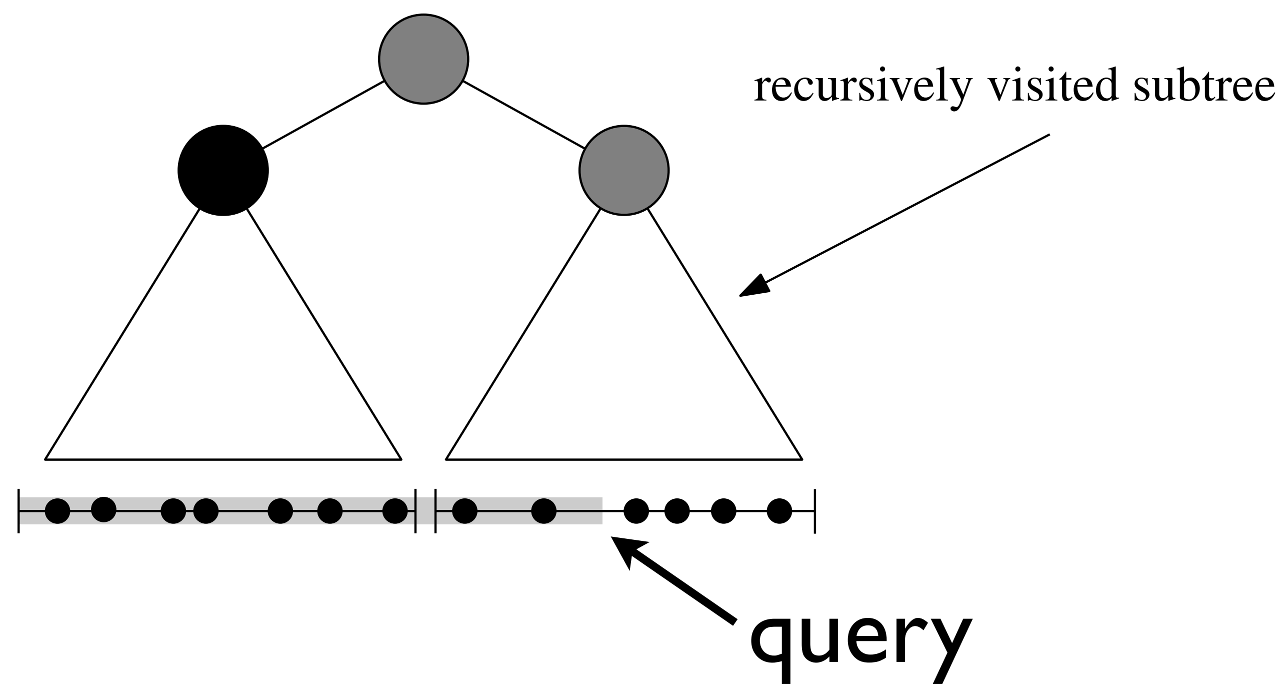


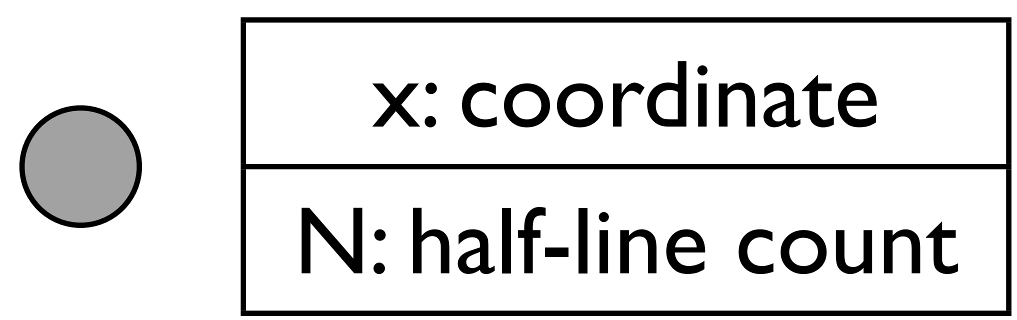
Searching Half Spaces

- Reduced complexity of our 2D range search from counting in simple polygons to counting in half-planes
- How can we quickly count points in half-spaces?
- Counting quickly in half-spaces is the motivation for partition trees

1-dimension

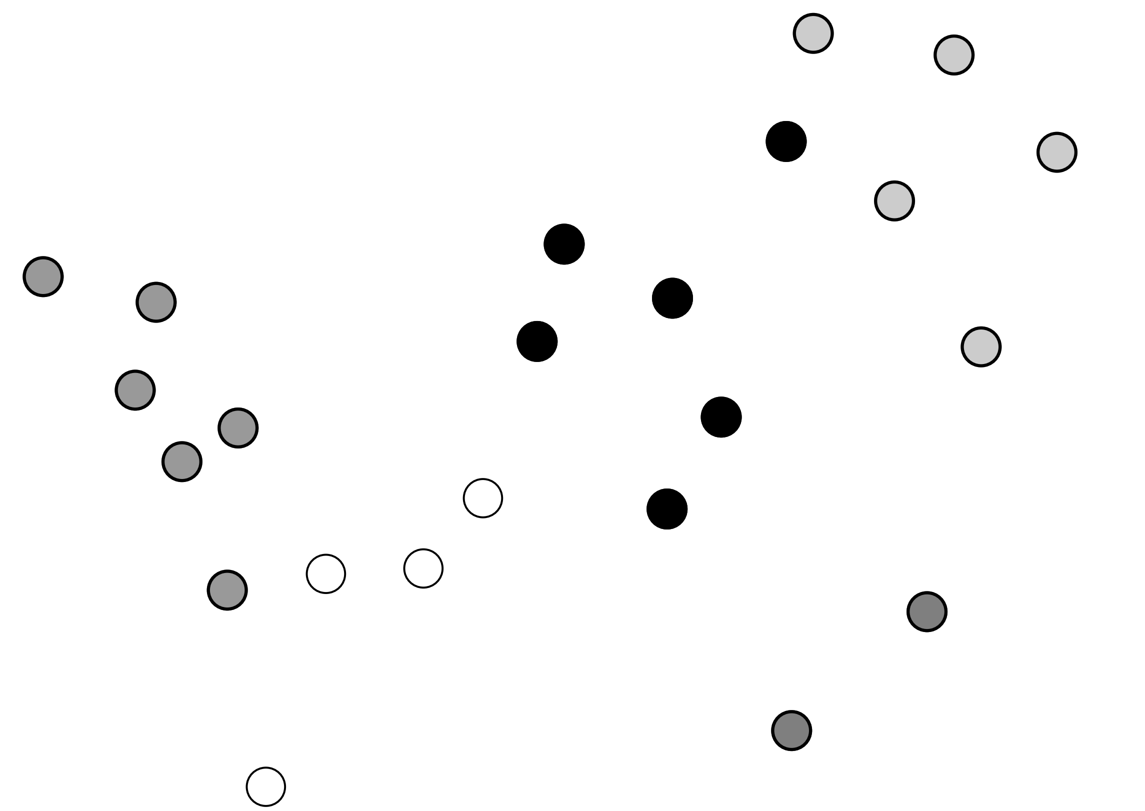
- Used a balanced binary search tree
- Child of root is two half-lines
- each node contains key-value pair (coordinate,count)
- At each level, must recurse on at most one subtree
- results in logarithmic query time





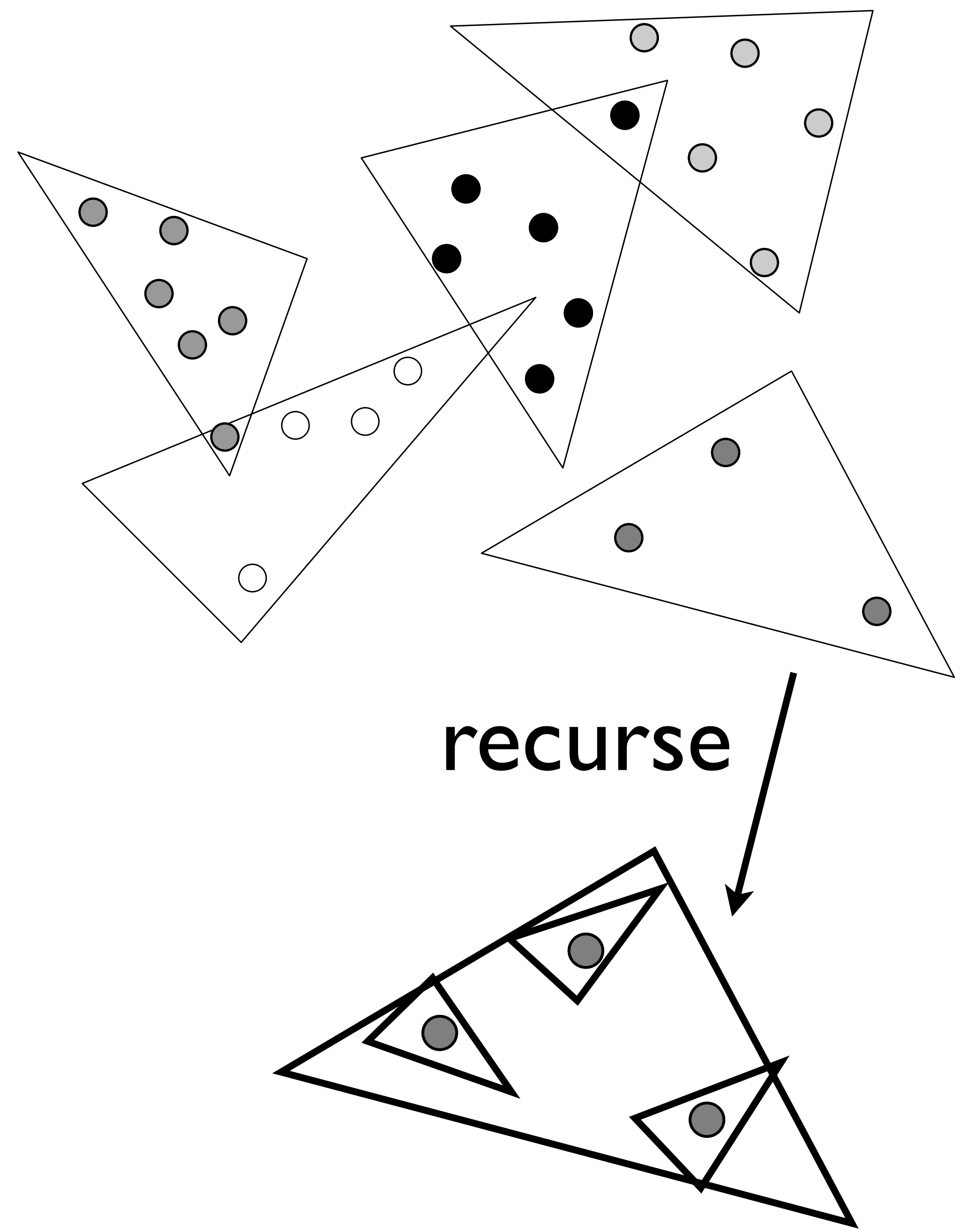
2-dimensions

- Not as easily partitioned as the line
- If root is entire plane, what do children represent?
- Second degree of freedom makes a partitioning scheme non-obvious



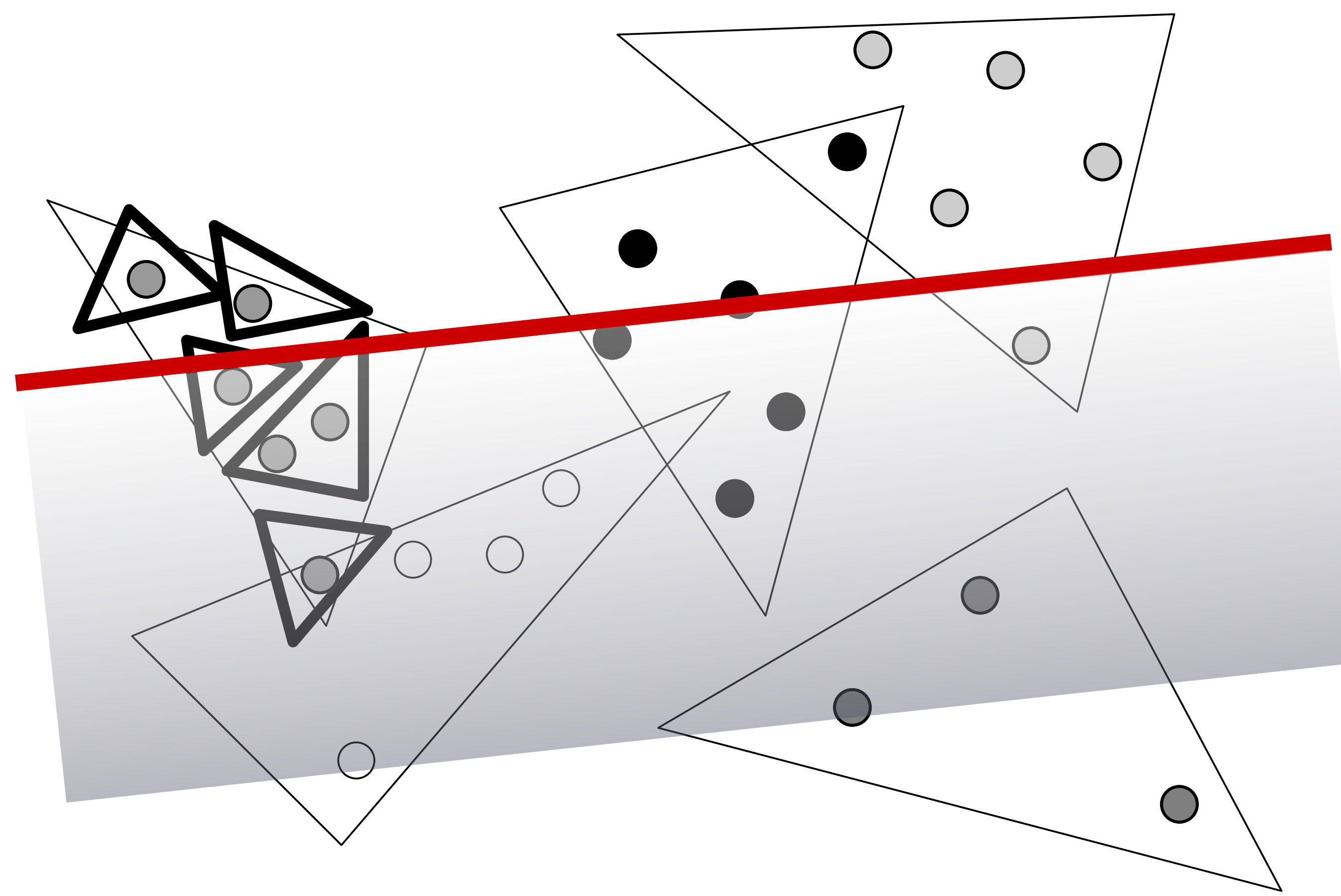
Simplical-partition

- Divide and conquer
- Partition point set with simplexes
- Simplexes can overlap in space, but each point can only be a member of one class
- Recursively partition simplexes

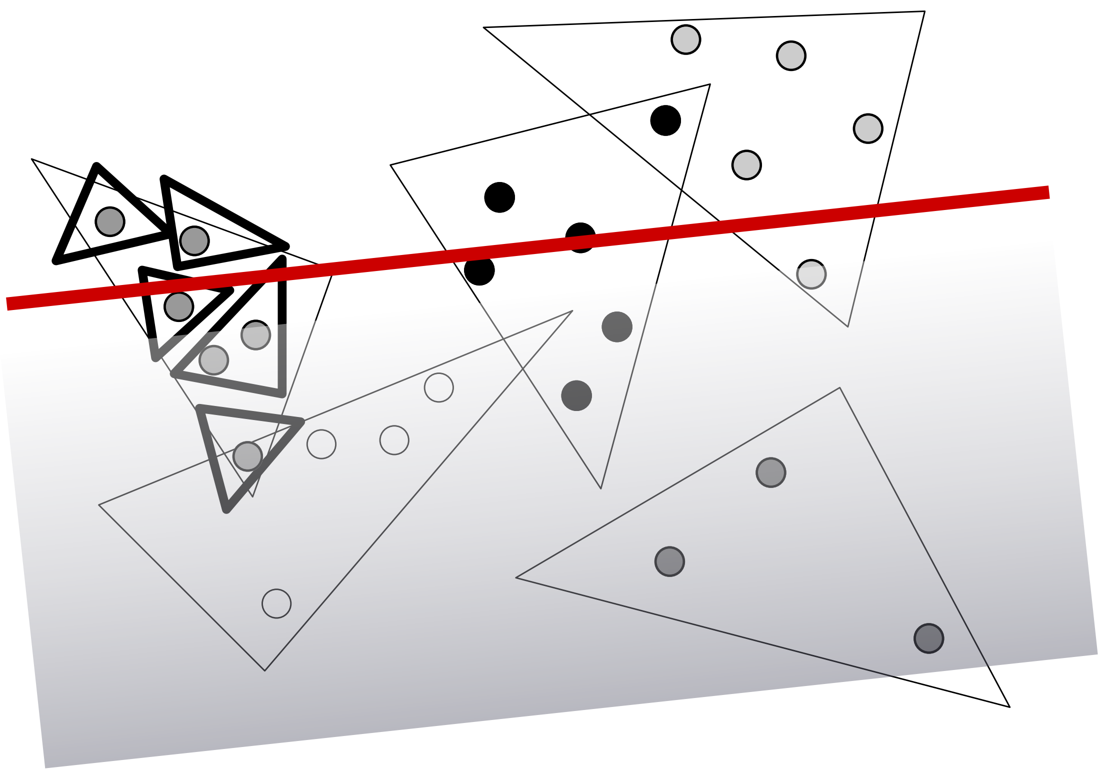


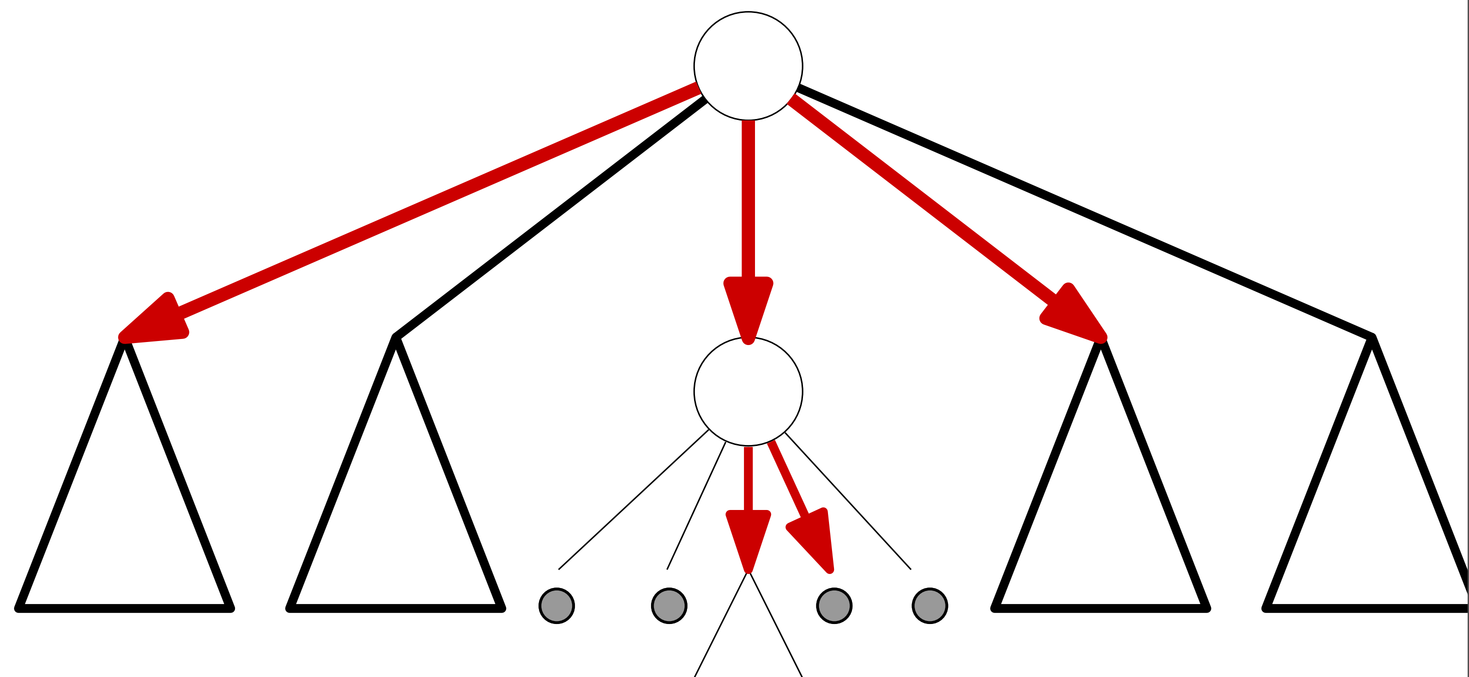
Half-plane counting

- Add points of triangles that are subsets of the half-plane
- If half plane crosses triangle, recurse into triangle and repeat test
- Recurse until all triangles lie on either side of half-plane or contain exactly one point



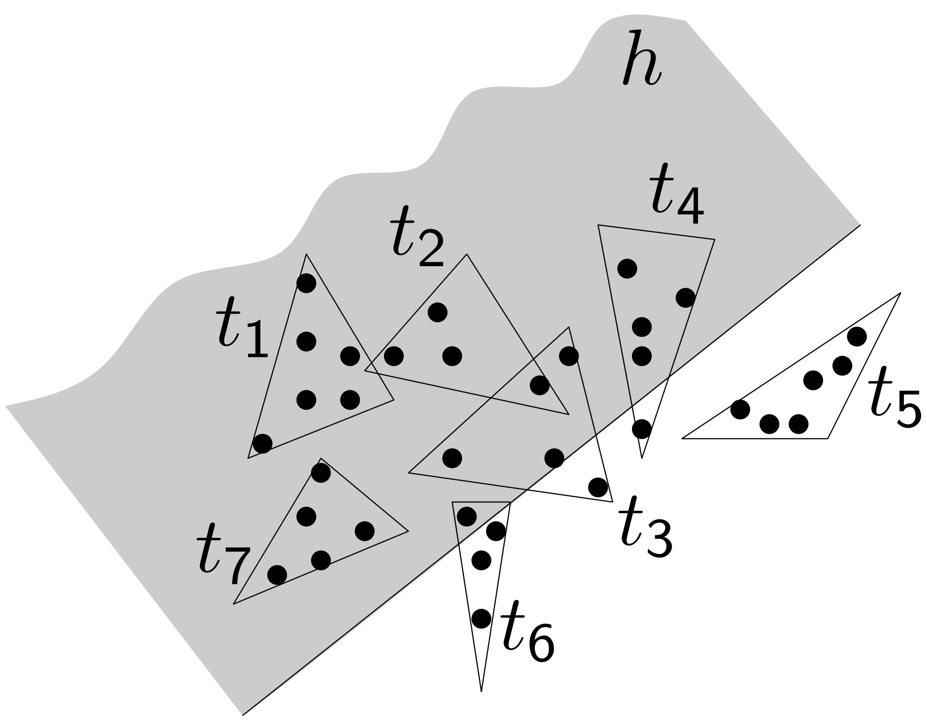
Partition Tree





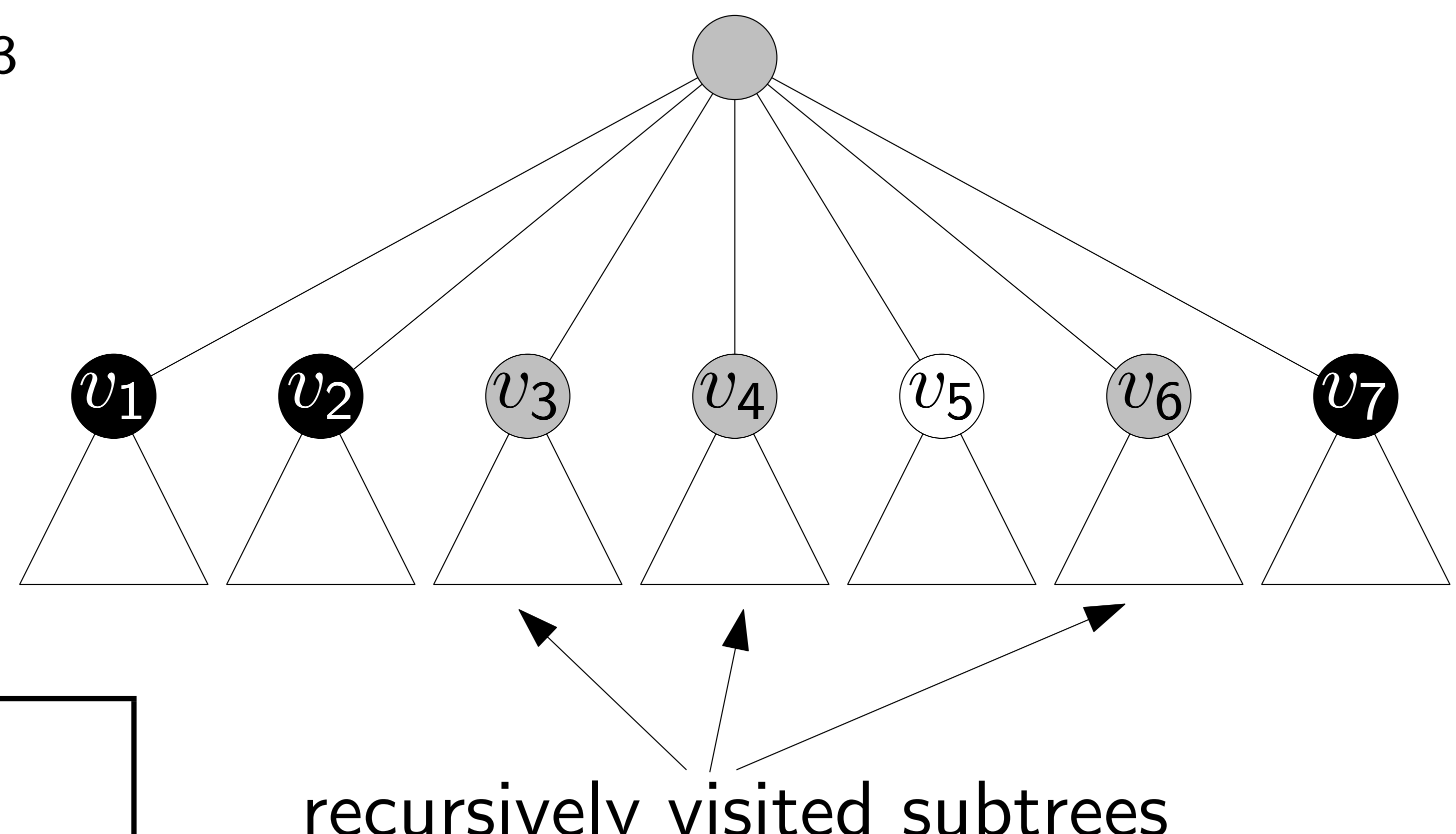
- Ternary tree where the number of children of any vertex corresponds to the number of simplexes needed to partition point set
- During query, subtrees are traversed if intersected by half plane. If m simplexes are intersected, m subtrees are traversed
- Each vertex of trees stores the simplex vertices and the number of interior points.

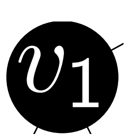
Range Query Example



● = selected node

● = visited node





S: simplex
N: simplex count

Range Query

$\textsc{SelectInHalfplane}$(half-plane h, partition tree $\mathcal{T}$)

$N \leftarrow \emptyset$ { set of *selected* nodes }

if $\mathcal{T} = \{\mu\}$ **then**

  if point stored at μ lies in h **then**

    $N \leftarrow \{\mu\}$

else

  for each child ν of the root of $\mathcal{T}$ **do**

    if $t(v) \subset h$ **then**

      $N \leftarrow N \cup \{\nu\}$

    else

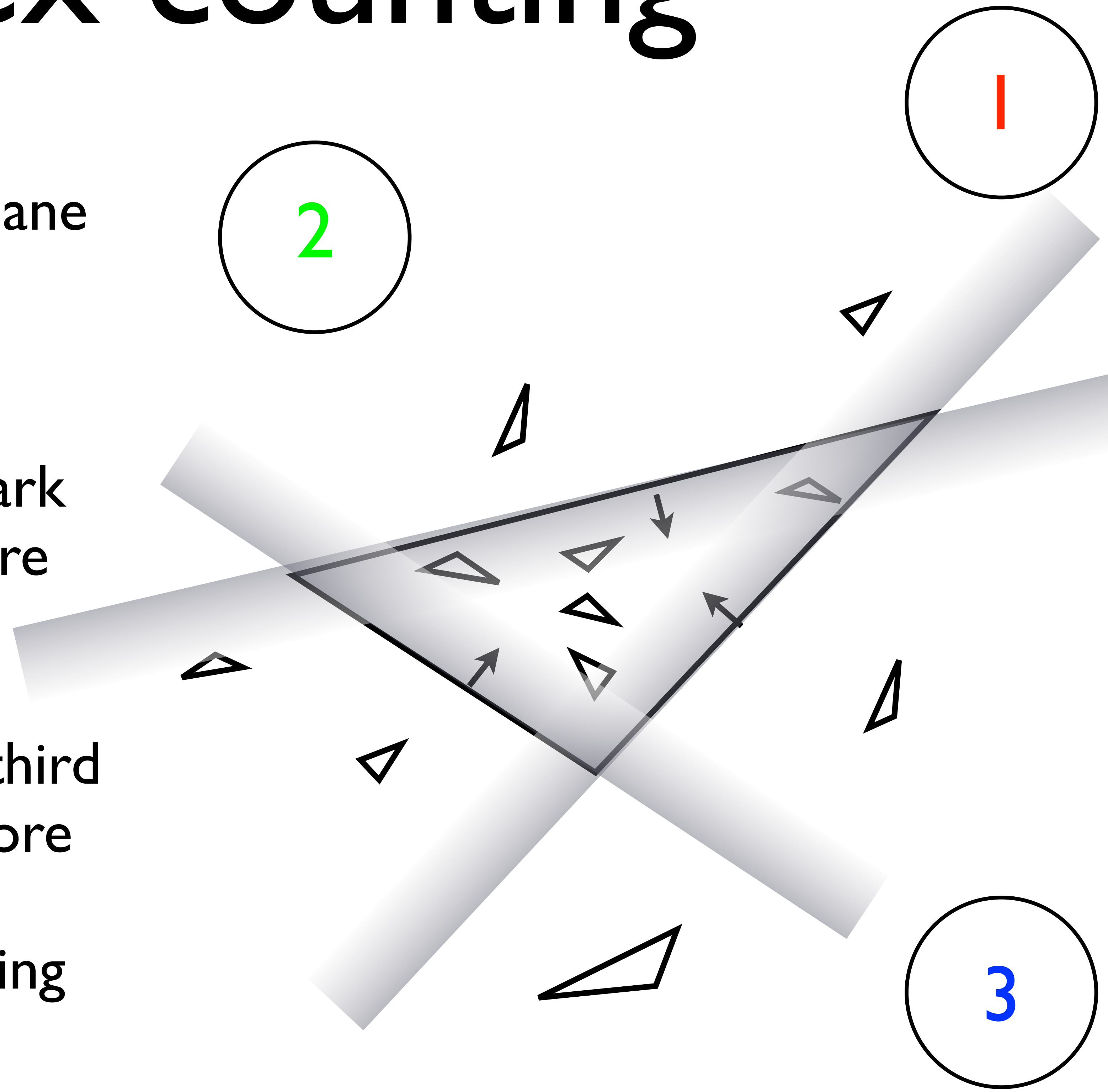
      if $t(\nu) \cap h \neq \emptyset$ **then**

        $N \leftarrow N \cup \textsc{SelectInHalfplane}(h, \mathcal{T}_\nu)$

return N

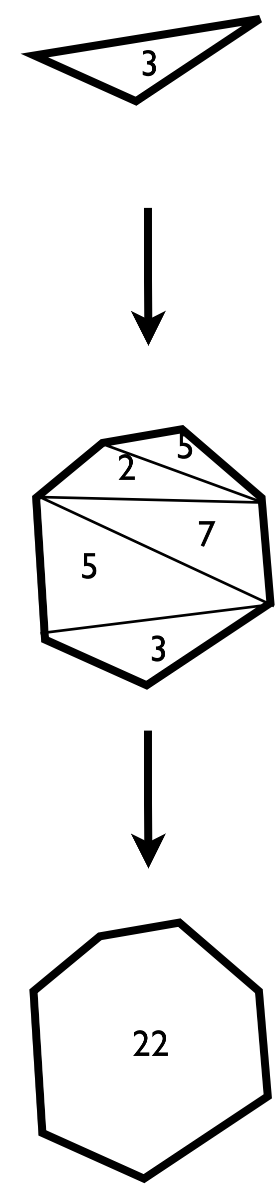
Simplex counting

- We conduct a half plane range query
- As we traverse the partition tree, we mark those vertices that are not in the half-plane
- On the second and third range search, we ignore previously marked vertices when counting



Region range search

- The sum the counts of all triangles is the region range search



Complexity

Half-Plane Query	Space	Construction
$O(\sqrt{n})$	$O(n)$	$O(n)$

Review

- Need to query on complex 2D regions, but that is hard
- Transform problem to sequence half-plane searches
- Bisecting half-spaces works in 1D, but does not generalize
- Need concept of recursive simplical partition to divide higher dimensional space
- Need of for fast half-plane searching on simplical partition motivates partition tree data structure

Complexity problem?

- Number of half-plane crossings is determined by the range query; query complexity is listed as $O(\sqrt{n})$. Why isn't this algorithm output sensitive life Jarvis?

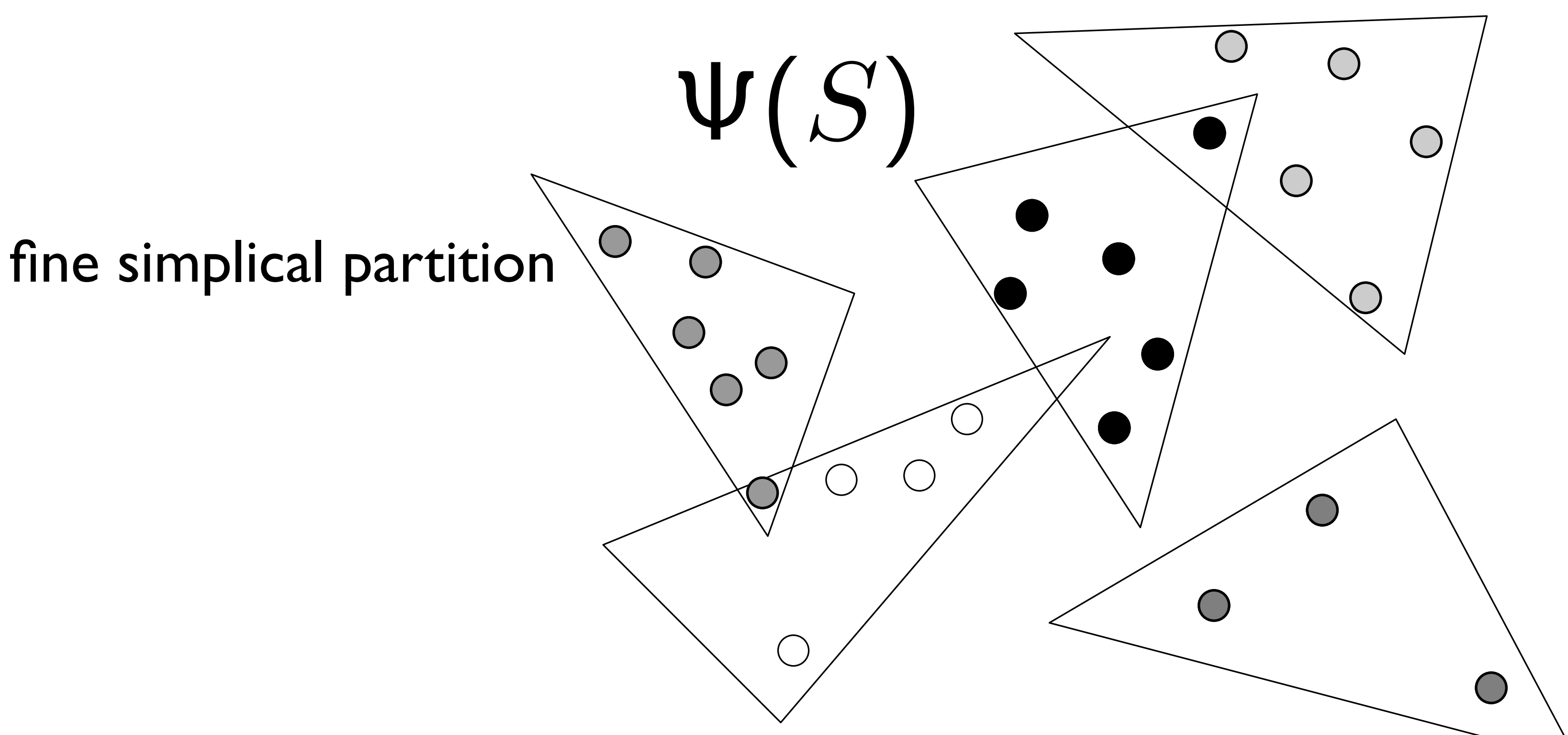
Fine simplical partitions

$\Psi(S) = \{(S_1, t_1), (S_2, t_2), \ldots, (S_r, t_r)\}$ is a *simplicial partition* (of size r) for S if

– S is partitioned by $S_1, \ldots, S_r$ and

– for $1 \leq i \leq r$, t_i is a triangle and $S_i \subset t_i$.

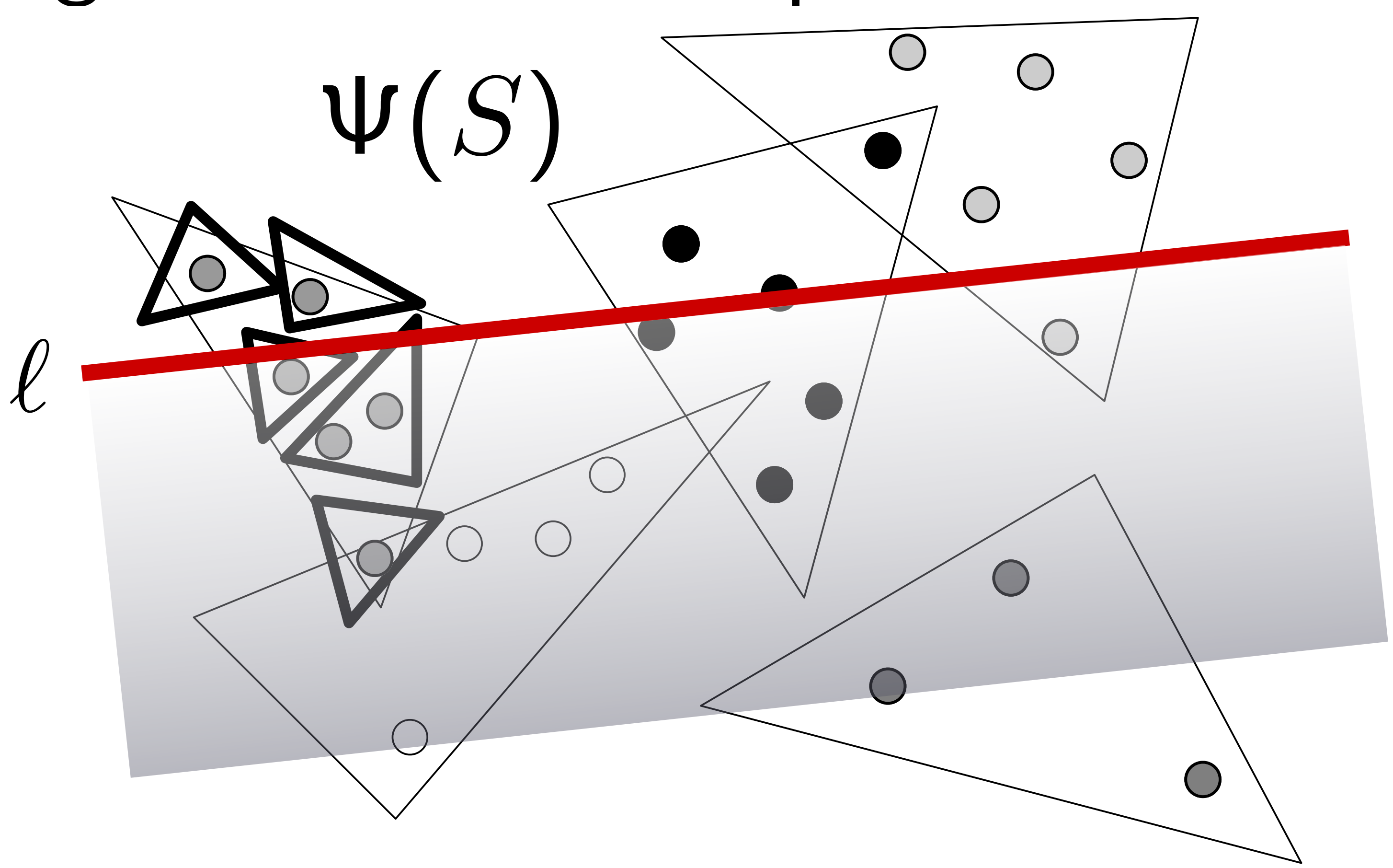
$\Psi(S)$ is *fine* if $|S_i| \leq 2|S|/r$ for every $1 \leq i \leq r$.



Crossing Numbers

The *crossing number* of ℓ (w.r.t. $\Psi(S)$) is the number of triangles $t_1, \ldots, t_r$ intersected by ℓ.

The *crossing number* of $\Psi(S)$ is the maximum crossing number over all possible lines.



Fine simplical construction

For any set S of n pts and any $1 \leq r \leq n$, a fine simplicial partition of size r and crossing number $O(\sqrt{r})$ exists.

For any $\varepsilon > 0$, such a partition can be built in $O(n^{1+\varepsilon})$ time.

Query complexity guarantees

For any $\varepsilon > 0$, there is a partition tree $\mathcal{T}$ for S s.t.:

for a query half-plane h, $\textsc{SelectInHalfplane}$
selects in $O(n^{1/2+\varepsilon})$ time
a set N of $O(n^{1/2+\varepsilon})$ nodes of $\mathcal{T}$
with the property that $h \cap S = \bigcup_{\nu \in N} S(\nu)$.

Half-plane range counting queries can be answered in $O(n^{1/2+\varepsilon})$ time.

Partition Tree construction

- Construct a fine simplical partition
  $\Psi(S) = \{(S_1, t_1), (S_2, t_2), \ldots, (S_r, t_r)\}$

- Add r vertices to partition tree with key-value pair (simplex, simplex count)

- Recurse on $S_1, \ldots, S_r$ and grow partition tree as in step 2

Fine simplical construction

- A topic for another presenter. I punt :)
- Two algorithms to construct fine-simplical construction: randomized and deterministic
- Second algorithm results directly from a inductive proof for the complexity bound of constructing an epsilon-cutting (see Matousek)
- Efficient partition trees formalized in mid 90s

Sources

Matoušek, J. Construction of ε-nets, *Disc. Comput. Geom.* 5 (1990), 427–448.

Matoušek, J. Efficient partition trees, *Disc. Comput. Geom.* 8 (1992), 315–334.

Chazelle, B., Sharir, M., Welzl, E. Quasi-optimal upper bounds for simplex range searching and new zone theorems, *Algorithmica* 8 (1992), 407–429.

De Berg, Mark; Cheong, Otfried; Van Kreveld, Marc; Overmars (2008). *Computational Geometry Algorithms and Applications*

Some images adapted from: Alexander Wolf's presentation for Geometric algorithms: http://www.win.tue.nl/~awolff/teaching/2009/2IL55/pdf/v16.pdf

Questions

- ?

OI-OPPA. European Social Fund

Prague & EU: We invest in your future.