Approximate Range Searching in the Absolute Error Model

Guilherme D. da Fonseca

CAPES BEX1319027

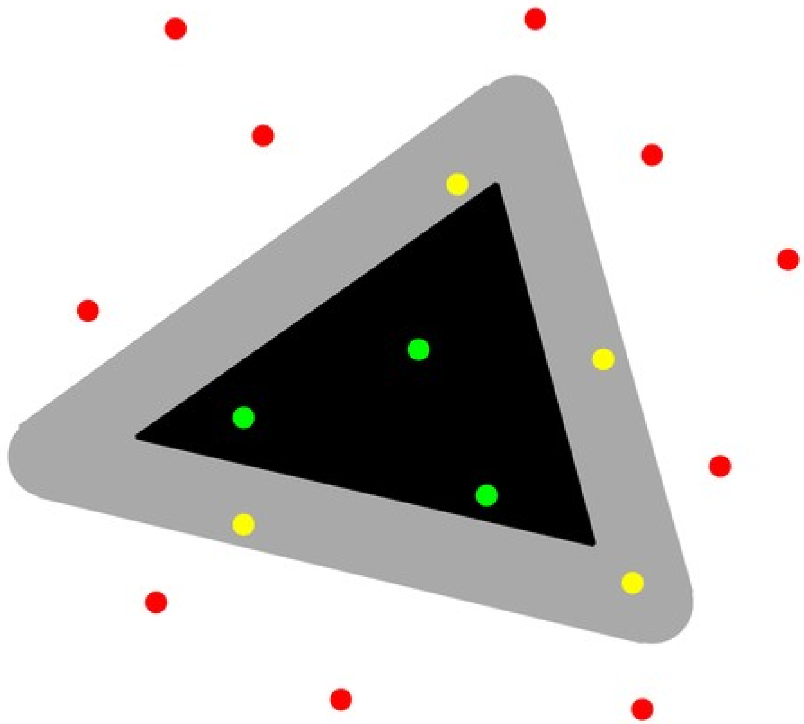
http://www.cs.umd.edu/~fonseca

Advisor: David M. Mount

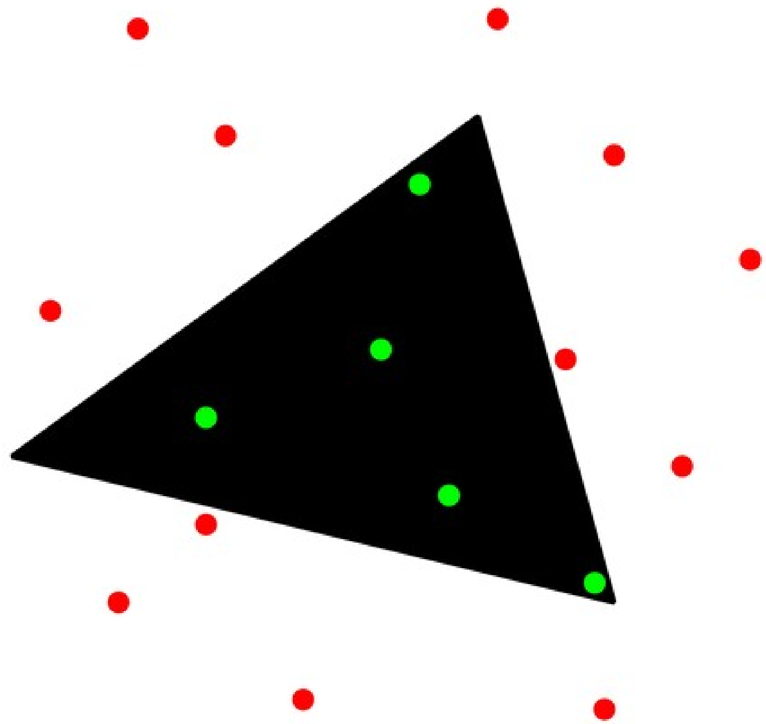
University of Maryland

WADS 2007

Contents



Range Searching – Exact Version



P: Set of n points in d-dimensional space.

w: Weight function.

$\mathcal{R}$: Set of *ranges* (regions of the space).

- Preprocess P, in a way that, given $R \in \mathcal{R}$, we can efficiently compute:

$$\sum_{p \in P \cap R} w(p)$$

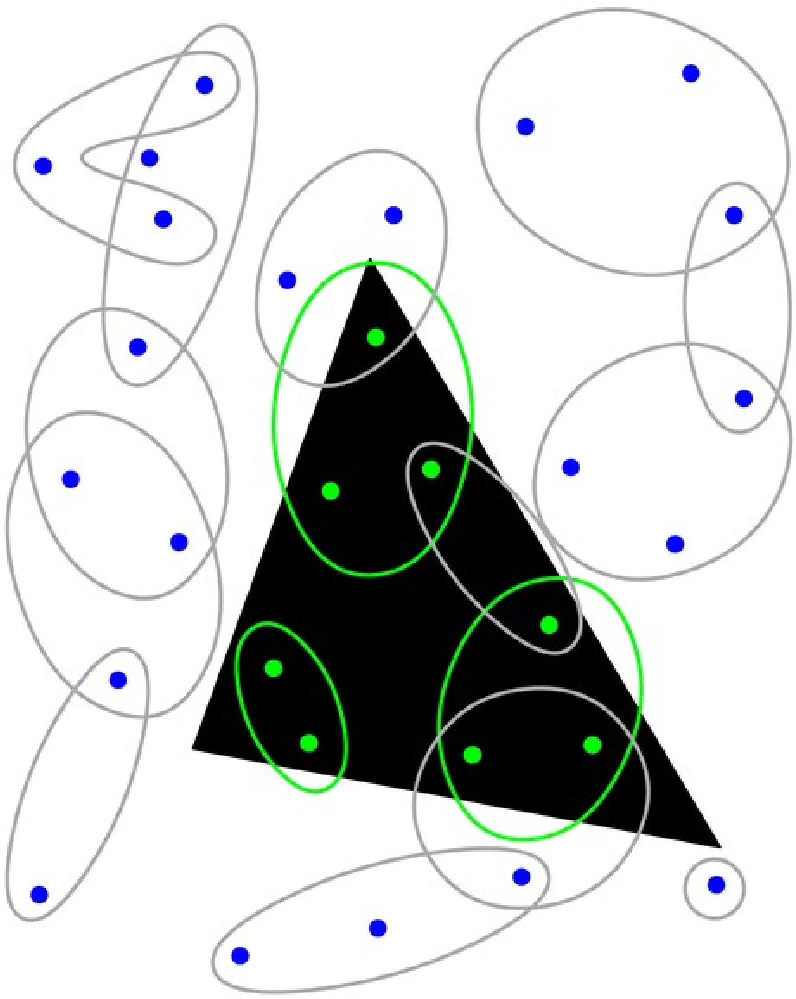
Semigroups, Groups and Idempotence

- In the most general version, the weights are drawn from a commutative **semigroup**.

Other properties may be useful:

- **Group**: We can use subtraction.
- **Idempotence**: For every x, we have $x+x=x$:
  - *maximum,*
  - Boolean *or*.

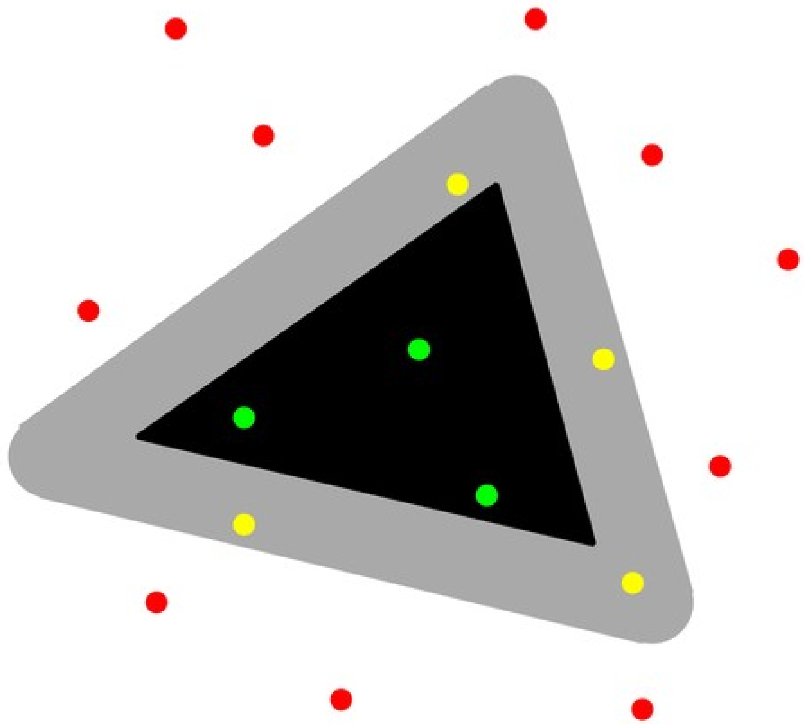
Generators



- Generators represent sets of points whose sum is precomputed.
- A query is processed by summing generators.

- Large generators:
  - Low query time,
  - High storage.
- Small generators:
  - High query time,
  - Low storage.

The Absolute Error Model

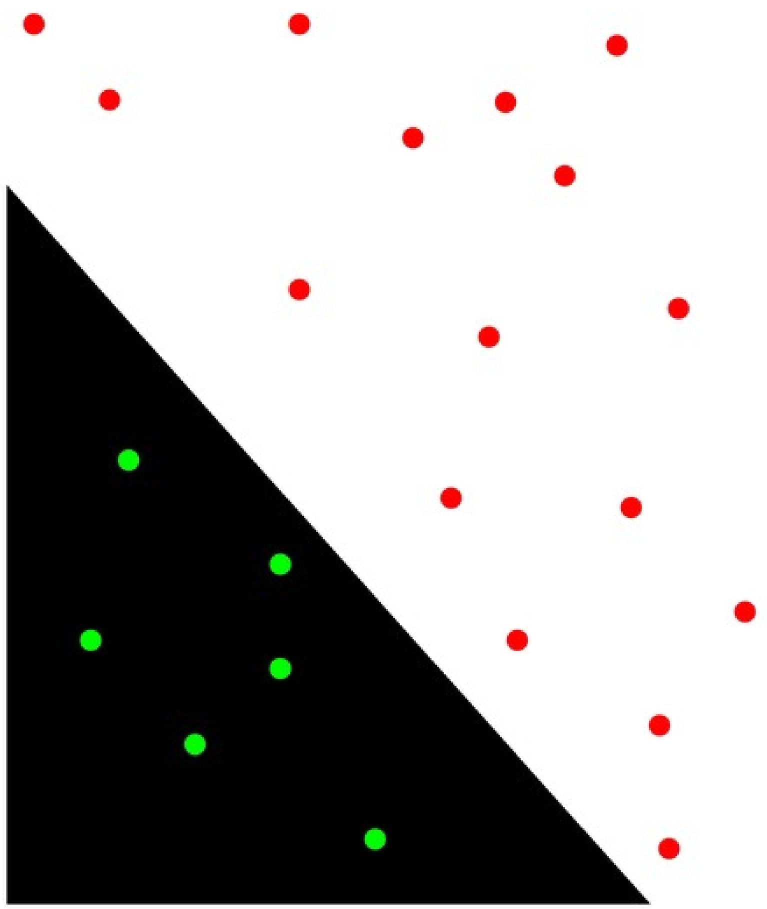


- **Absolute** error model: points within distance ε from the boundary may be counted or not
- All data points lie inside the unit hypercube $[0,1]^d$.
- **Relative** error model: fuzzy boundary is proportional to the diameter of the range (Arya, Mount, 2000).

Our Results

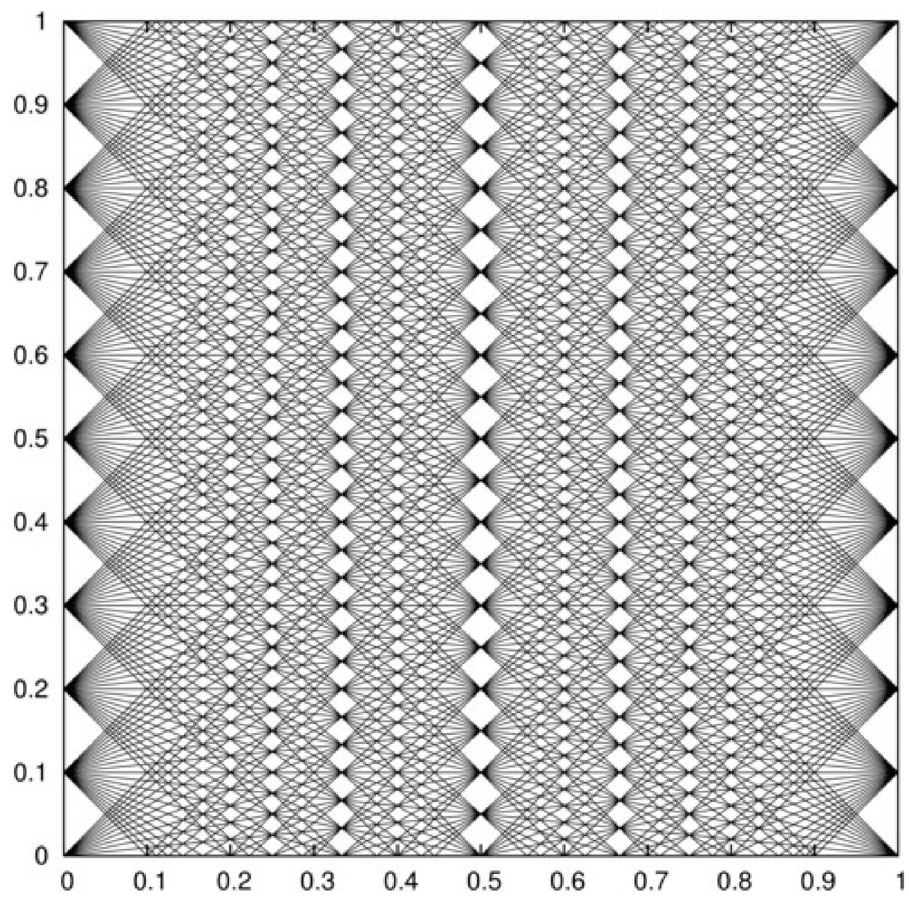
- The first work on approximate range searching in the **absolute error model** for fixed dimensions.
- Our data structures are **simple** and amenable to **efficient implementation**.
- We exploit **idempotence** to achieve better performance.
- Optimal “data structure” for **exact** idempotent halfspace range searching.
- Introduction of the versatile **halfbox quadtree**.

Halfspace Ranges



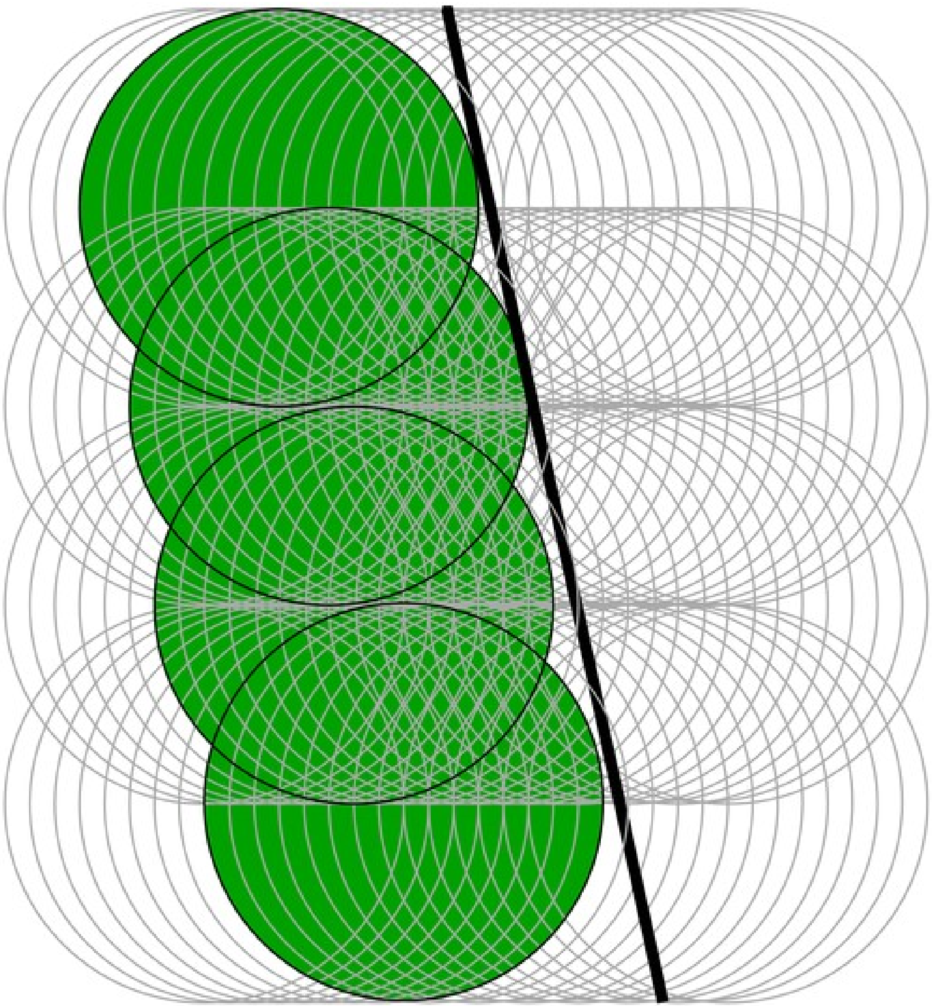
- Ranges are d-dimensional halfspaces.
- Exact (Matoušek, 1993):
  - Query time: O($n^{1\text{-}1/d}$).
  - Space: O(n).
  - Polylogarithmic query time takes n^d space.
- Approximate:
  - Query time: O(1).
  - Space: O(1/ε^d).

Halfspace Ranges Structure



- We can ε-approximate all halfspaces inside the unit cube with $O(1/\varepsilon^d)$ halfspaces.
- Store the results in a table.
- Naive preprocessing takes $O(n/\varepsilon^d)$ or $O(n+1/\varepsilon^{2d})$ time.
- We show how to use $\tilde{O}(n+1/\varepsilon^d)$ preprocessing time.

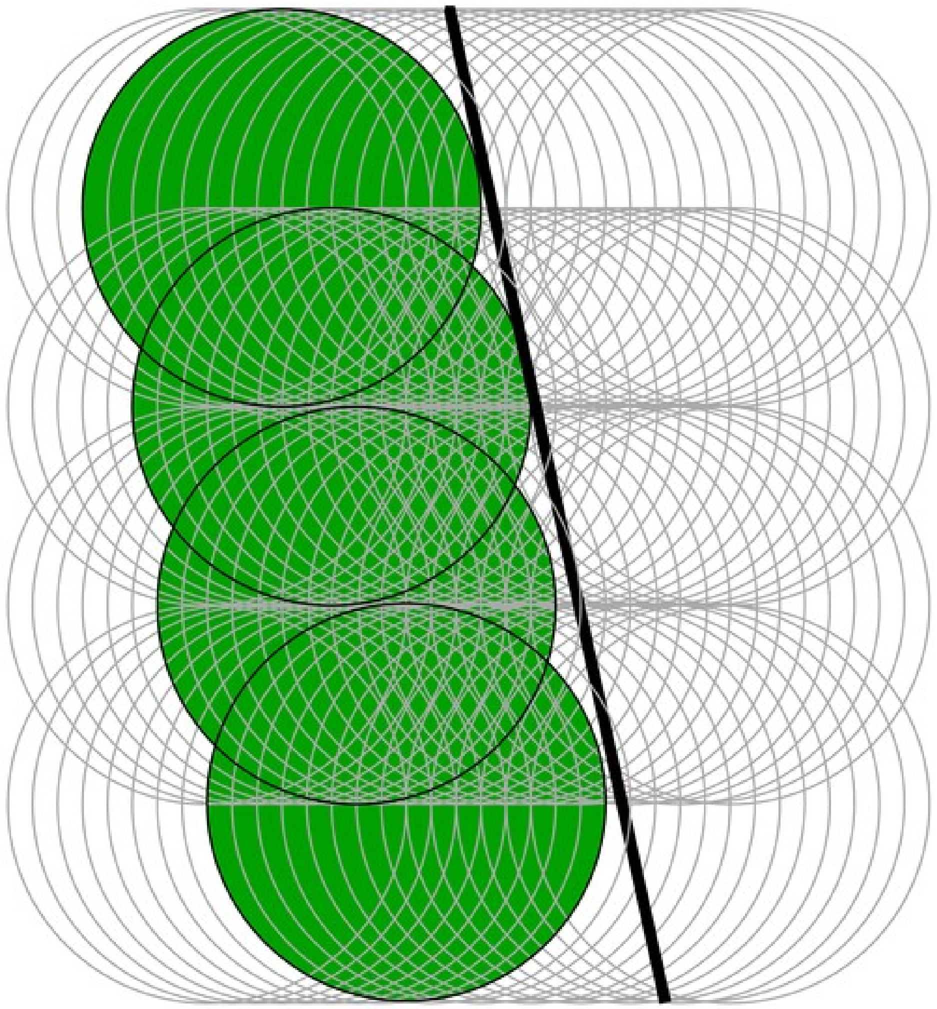
Idempotent Version



- Idempotent semigroup: $x+x=x$, for all x.
- Generators can overlap.
- Use large spherical generators.
  - Space: $O(1/\varepsilon^{(d+1)/2})$.
  - Query time: $O(1/\varepsilon^{(d-1)/2})$.
  - Trade-off: $O(m)$ space, $O(1/m\varepsilon^{d})$ query time.

Exact Uniformly Distributed Idempotent Version

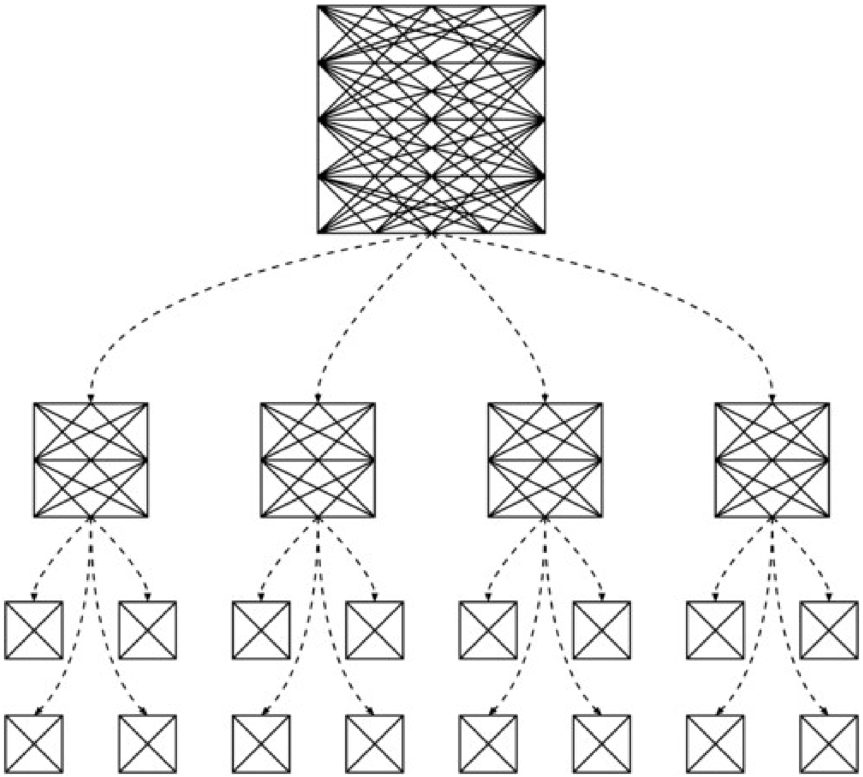
- Approximate version:
  - Space: $O(1/\varepsilon^{(d+1)/2})$.
  - Query time: $O(1/\varepsilon^{(d-1)/2})$.
- Set $\varepsilon = 1/n^{2/(d+1)}$:
  - Space: $O(n)$.
  - Query: $O(n^{1-2/(d+1)})$.
- The $\varepsilon n = O(n^{1-2/(d+1)})$ remaining points are counted one by one.



Exact Uniformly Distributed Idempotent Version

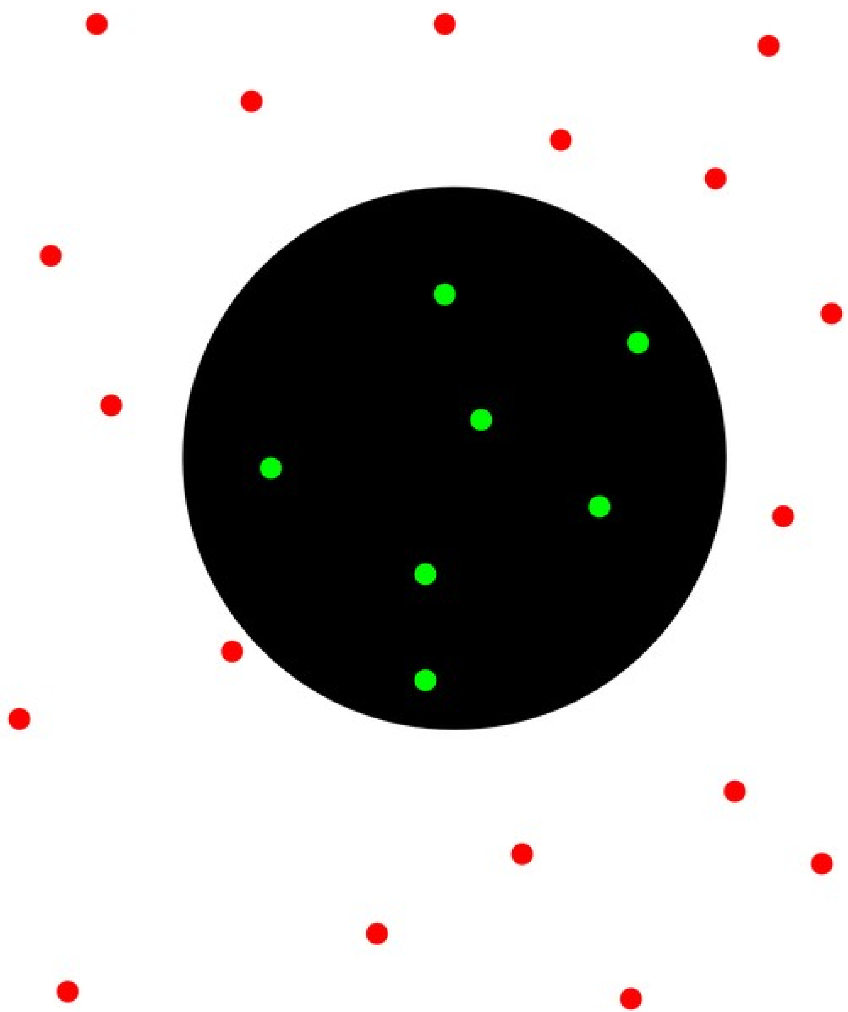
- Approximate version:
  - Space: $O(1/\varepsilon^{(d+1)/2})$.
  - Query time: $O(1/\varepsilon^{(d-1)/2})$.
- Set $\varepsilon = 1/n^{2/(d+1)}$:
  - Space: $O(n)$.
  - Query: $O(n^{1-2/(d+1)})$.
- The $\varepsilon n = O(n^{1-2/(d+1)})$ remaining points are counted one by one.
- Works in the *semigroup arithmetic model*.
- Uniform distribution.
- Matches the best lower bound up to logarithmic terms (Brönnimann, Chazelle, Pach, 1993).
- Same assumptions as the lower bound.

Halfbox Quadtree



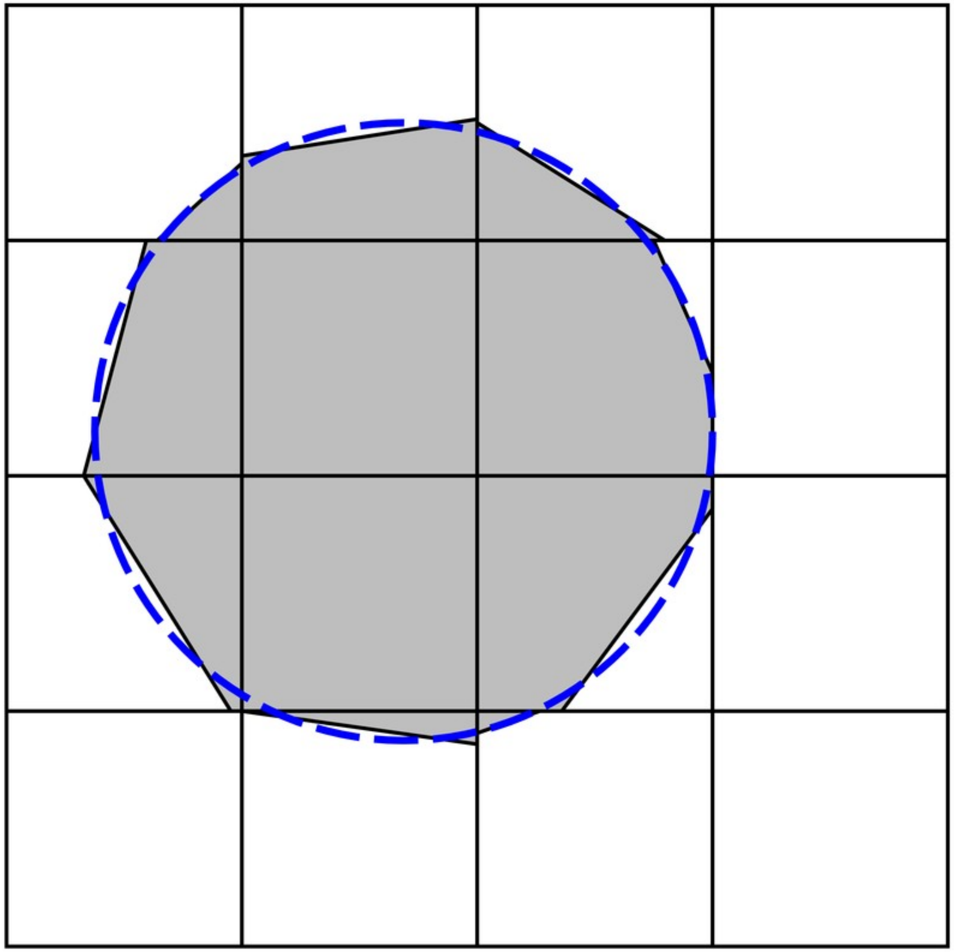
- One halfspace data structure for each quadtree box.
- Generators: intersection of quadtree boxes and halfspaces (*halfboxes*).
- Powerful building blocks!
- Smaller boxes take less space, as ε is constant.
- Storage space is $O(\log(1/\varepsilon)/\varepsilon^d)$.

Spherical Range Searching



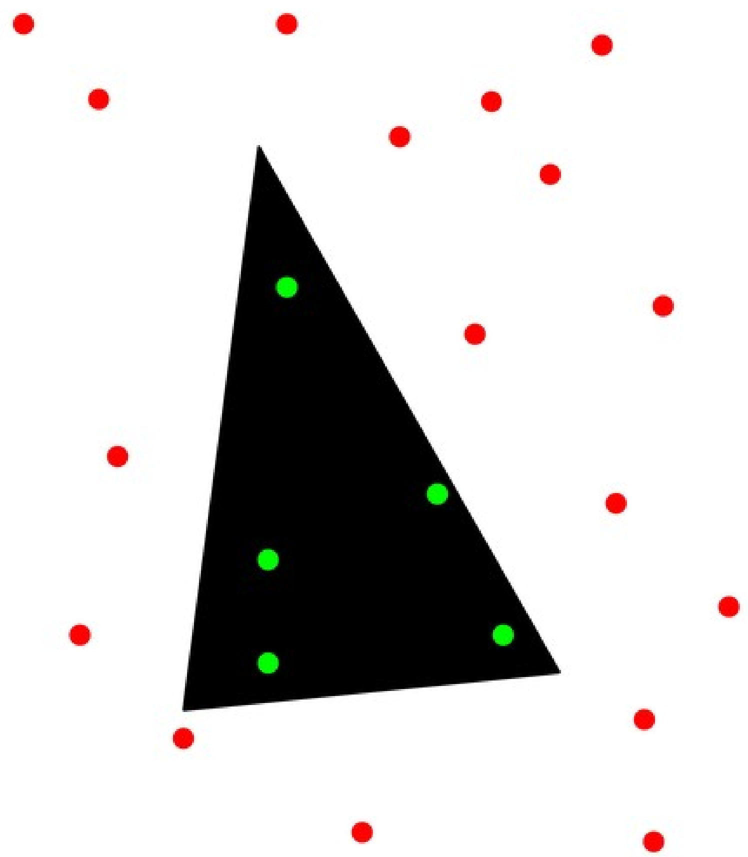
- Ranges are d-dimensional spheres.
- Exact version:
  - Project the points onto a $(d+1)$-dimensional paraboloid.
  - Use halfspace range searching.
- Approximate version:
  - Use the halfbox quadtree.

Spherical Range Searching



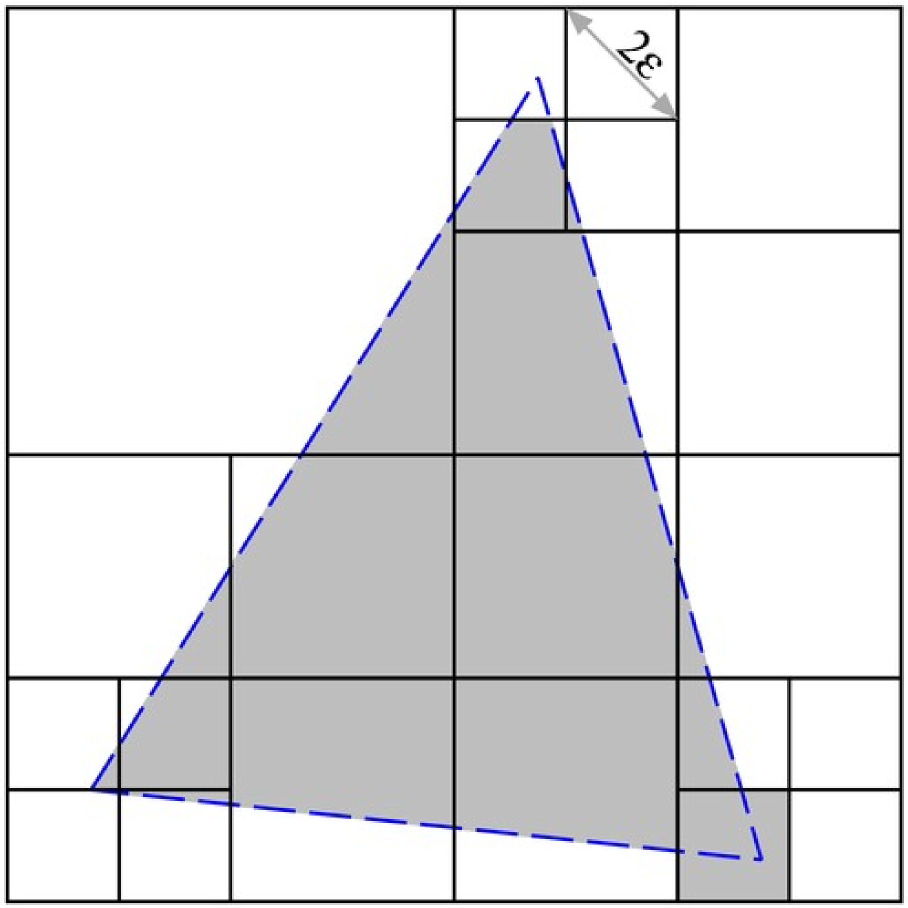
- Approximate the range with halfboxes.
- Only $O(1/\varepsilon^{(d-1)/2})$ halfboxes are necessary.
- Use the halfbox quadtree to query each halfbox in $O(1)$ time.
  - Query time: $O(1/\varepsilon^{(d-1)/2})$.
  - Space: $O(\log(1/\varepsilon)/\varepsilon^{d})$.

Simplex Range Searching



- Ranges are d-dimensional simplices: intersection of d halfspaces.
- Exact version solved similarly to halfspace range searching: (Matoušek, 1993)
  - Query time: $O(n^{1-1/d})$.
  - Space: $O(n)$.

Simplex Range Searching



- Use the halfbox quadtree.
- Recurse when you hit a (d-2)-face.
- Otherwise, subtract all disjoint (d-1)-faces.
- Group version:
  - Query time: $O(1/\varepsilon^{d-2} + \log(1/\varepsilon))$.
  - Space: $O(\log(1/\varepsilon)/\varepsilon^{d})$.

Conclusions

- The absolute error model is better suited for several applications.
- Data structures are simpler than exact and other approximate data structures.
- Halfspace range searching is extremely fast.
- In the idempotent version, halfspace range searching requires little space (with slower query time).
- The halfbox quadtree is efficient for various shapes of ranges.

Future Research

- Improved data structures? Lower bounds?
- How much space is required to achieve O(1) query time for different shapes of ranges?
- Which other problems are interesting in the absolute error model?
- Which other models of computation? (Data stream model, for example.)
- Better understanding of the connection between exact, relative approximation, and absolute approximation.