# Fast Preprocessing for Optimal Orthogonal Range Reporting and Range Successor with Applications to Text Indexing

Younan Gao & Meng He

Dalhousie University

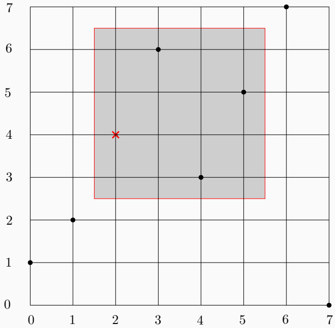
Yakov Nekrich

Michigan Technological University

# Three Orthogonal Range Searching Problems

Three orthogonal range searching problems are

- **Counting**: compute the number of points contained in $N \cap Q$;
- **Reporting**: report the points in $N \cap Q$;
- **Successor**: retrieve the point in $N \cap Q$ with the smallest $x$- or $y$-coordinate;



# Motivation

- Orthogonal range searching has many applications:
  - ◇ text indexing:

    Mäkinen and Navarro LATIN'2006

    Chien at al. Algorithmica'2015

    Munro at al. CPM'2020
  - ◇ Lempel-Ziv factorization

    Belazzougui and Puglisi SODA'2016
  - ◇ constructing Adam consensus tree

    Jansson et al. STACS'2015
- Preprocessing time is often **neglected**, but it matters:
  - ◇ As a building block of an algorithm processing plain data;
  - ◇ As components of text indexes.

# Previous Work

| Breaking the $O(n \lg n)$ Bound on the Construction Time | | |
|---|---|---|
| | **Query Time** | **Space Cost** |
| Counting | $O(\frac{\lg n}{\lg \lg n})$ | $O(n)$ words |
| Building Wavelet Tree | - | $O(n \lg \sigma)$ bits |
| Successor | $O(\lg^{\epsilon} n)$ | $O(n)$ words |
| | **Construction Time** | **Citation** |
| Counting | $O(n\sqrt{\lg n})$ | Chan and Pătraşcu SODA'2010 |
| Building Wavelet Tree | $O(\frac{n \lg \sigma}{\sqrt{\lg n}})$ | Babenko et al. SODA'2015 |
| Successor | $O(n\sqrt{\lg n})$ | Belazzougui and Puglisi SODA'2016 |

| Sorted reporting with $O(n \lg n)$ preprocessing time | | |
|---|---|---|
| **Query Time** | **Space Cost** | **Citation** |
| $O((\boldsymbol{occ}+1) \lg \lg n)$ | $O(n \lg \lg n)$ words | Zhou Inf. Process. Lett.'2016 |
| $O(\lg \lg n + \boldsymbol{occ})$ | $O(n \lg^{\epsilon} n)$ | Nekrich and Navarro SWAT'2012 |

**Challenge:** 2d orthogonal range reporting or range successor requires superlinear space.

# Preliminaries

Word Ram model:

- Word size $w = \Theta(\lg n)$ bits;
- Operations on words taking constant time;
- Nonstandard operations can be simulated by table lookup of $o(n)$ bits

*Packed Sequence* $A \in [\sigma]^{n'}$ *:*

- Space: $\lceil n' \lceil \lg \sigma \rceil / w \rceil$ words;
- Pre-processing Time: Improved from $O(n')$ to $O(n' \lg \sigma / \lg n)$ for `rmq`/`rMq` and `succ`/`pred` quereis.

# $\mathit{rank}'$ Queries with Fast Construction

**Definition:** A partial rank operation, $\mathit{rank}'(A, i)$, computes the number of elements equal to $A[i]$ in $A[0..i]$.

| i: | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A: | a | a | b | c | d | d | c | b | a | b | b | c |
| $\mathit{rank}'(A, i)$: | 1 | 2 | 1 | 1 | 1 | 2 | 2 | 2 | 3 | 3 | 4 | 3 |

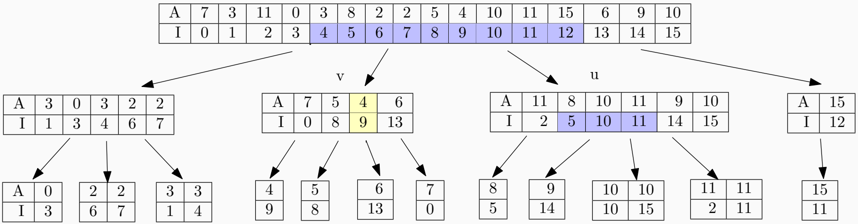
**Our Result:** For a packed sequence $A[0..n'-1]$ drawn from alphabet $[\sigma]$, where $n' \leq n$ and $\sigma = O(2^{O(\sqrt{\lg n})})$:

- Space: $n'\lceil \lg \sigma \rceil + o(n' \lg \sigma)$ extra bits;
- Pre-processing Time: $O(n' \lg^2 \sigma / \lg n + \sigma)$;
- $O(1)$ time and $O(1)$ accesses to elements of $A$;

# Definition of the Ball Inheritance Structure

The *ball inheritance* problem is defined over a range tree $T$ to support:

- `point`$(v, i)$, which returns the point $(A(v)[i], I(v)[i])$ in $N$ for an arbitrary node $v$ in $T$ and an integer $i$; and
- `noderange`$(c, d, v)$, which, given a range $[c, d]$ and a node $v$ of $T$, finds the range $[c_v, d_v]$ such that $I(v)[i] \in [c, d]$ iff $i \in [c_v, d_v]$.



# Improvements on building ball inheritance structures

**Input:** A sequence $A[0..n'-1]$ drawn from $[\sigma]$;

**Output:** A $d$-ary wavelet tree augmented with ball inheritance data structure

**Previous Results:** Chan et al. SoCG'2011

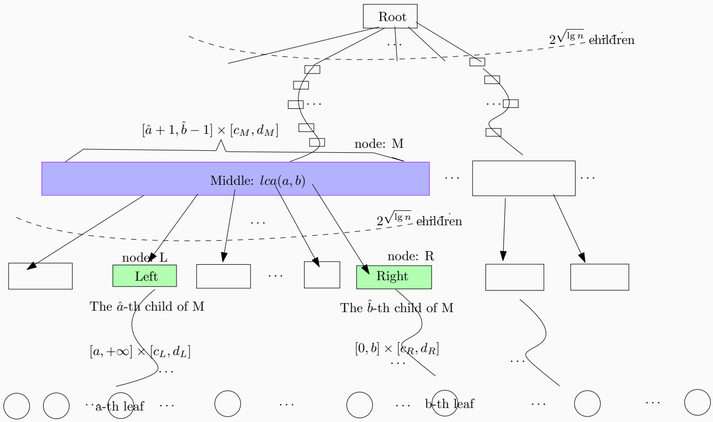
| Space Cost | `point` | `noderange` | Preprocessing |
|---|---|---|---|
| $O(n' \lg n')$ | $O(\lg^{\epsilon} \sigma)$ | $O(\lg\lg n' + \lg^{\epsilon} \sigma)$ | |
| $O(n'(\lg n') \lg\lg \sigma)$ | $O(\lg\lg \sigma)$ | $O(\lg\lg n' + \lg\lg \sigma)$ | $O(n' \lg n')$ |
| $O(n'(\lg n') \lg^{\epsilon} \sigma)$ | $O(1)$ | $O(\lg\lg n')$ | |

**Our Improvements:** when $\sigma = O(2^{O(\sqrt{\lg n})})$ and $n' = O(\sigma^{O(1)})$

| Space Cost | `point` | `noderange` | Preprocessing |
|---|---|---|---|
| $O(n' \lg \sigma \lg\lg \sigma + \sigma w)$ | $O(\lg\lg \sigma)$ | $O(\lg\lg \sigma)$ | $O(n' \lg^2 \sigma / \lg n +$ |
| $O(n' \lg \sigma \log_d^{\epsilon} \sigma + \sigma w)$ | $O(1)$ | $O(\lg\lg \sigma)$ | $\sigma \log_d \sigma)$ |

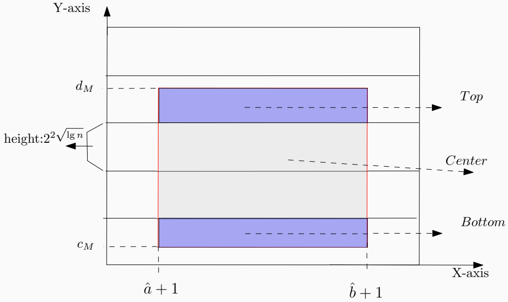
# The Framework of Orthogonal Range Reporting

- A $2^{\sqrt{\lg n}}$-ary wavelet tree built over $x$-coordinates of points;
- The tree height is $O(\sqrt{\lg n})$;
- Nodes: $M = \mathtt{lca}(\mathtt{leaf}_a, \mathtt{leaf}_b)$, $L/R = \mathtt{child}(M)$ on the path from $M$ to $\mathtt{leaf}_a/\mathtt{leaf}_b$



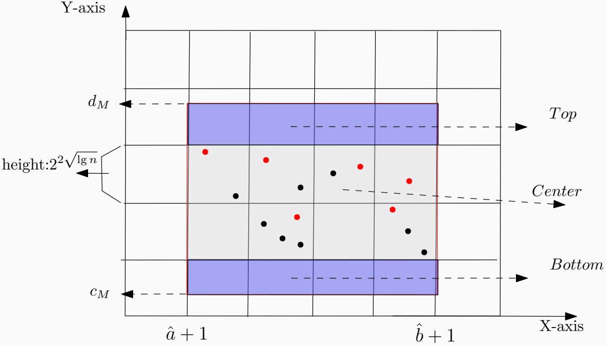
# Range Reporting over the Middle Part

- Each $2^{2\sqrt{\lg n}}$ consecutive points along Y-axis are assigned into a chunk:
- The query range $[\hat{a}+1, \hat{b}-1] \times [c_M, d_M]$ is divided into three non-overlapping parts: **Bottom, Center and Top**;
- **Bottom/Top**: 4-sided range reporting in a $2^{\sqrt{\lg n}} \times 2^{2\sqrt{\lg n}}$ grid:
  - ◇ The fast-built ball inheritance structure under the special conditions can apply;
  - ◇ *rmq*/*rMq* queries over packed sequences



# Range Reporting over the Center Part

- **Sampled Points:** Sampling any point from each of the $2^{\sqrt{\lg n}} \times |N(M)|/2^{2\sqrt{\lg n}} = |N(M)|/2^{\sqrt{\lg n}}$ cells;
- **Structure:** 4-sided reporting over selected points with $O(|N(M)|/2^{\sqrt{\lg n}} \lg^{O(1)} n) = o(|N(M)|)$ pre-processing time;
- **An observation:** If a sampled point $p$ is in the query range, all points in the same cell that $p$ belongs to are also in the query range.



# Our Results

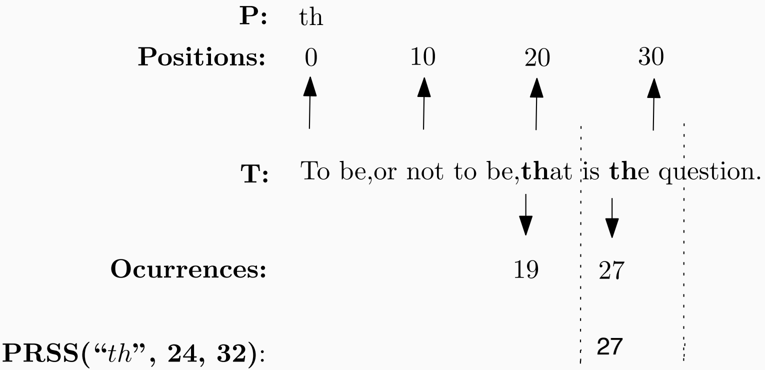
| | Query Time | Space Cost (words) | Construction Time |
|---|---|---|---|
| Reporting | $O(\lg\lg n + \boldsymbol{occ})$ | $O(n \lg^{\epsilon} n)$ | $O(n\sqrt{\lg n})$ |
| Successor | $O(\lg\lg n)$ | $O(n \lg\lg n)$ | $O(n\sqrt{\lg n})$ |
| Sorted Reporting | $O(\lg\lg n + \boldsymbol{occ})$ | $O(n \lg^{\epsilon} n)$ | $O(n\sqrt{\lg n})$ |

- For $n$ points in 2d rank space;
- More levels of reductions are required for orthogonal range successor and sorted range reporting;

# Text Indexing and Searching in Sublinear Time

**Definition:** Preprocessing a text string $T \in [\sigma]^n$, such that, given a pattern string $P[0..p-1]$,

- **Listing Queries:** one can report all these occurrences in $T$;
- **Position-Restricted Substring Searches (PRSS):** one can report all these occurrences in $T[l..r]$, where $l$ and $r$ are two indices.



# Our Results

| Listing Queries | | |
|---|---|---|
| Query | $O(p/\log_\sigma n + \log_\sigma n \lg\lg n + occ)$ | |
| Space | $O(n \lg\sigma \lg^\epsilon n)$ bits | |
| | Munro at al. 2020 | **New** |
| Preprocessing | $O(n \lg\sigma \lg^\epsilon n)$ | $O(n \lg\sigma/\sqrt{\lg n})$ |

| Position-Restricted Substring Searches | | |
|---|---|---|
| Space | $O(n \lg^{1+\epsilon} n)$ bits | |
| | Bille and Gørtz 2014 | **New** |
| Preprocessing | $O(n \lg n)$ expected | $O(n\sqrt{\lg n})$ |
| Query | $O(p + occ)$ | $O(p/\log_\sigma n + \lg p + \lg\lg\sigma + occ)$ |

# Conclusion

- Fast Pre-processing for 2-d orthogonal range search problems:
  - Orthogonal range reporting;
  - Orthogonal range successor;
  - Orthogonal sorted range reporting;
- The same space costs and query time of the previous best tradeoffs;
- Improvements on Text indexing
  - Listing queries;
  - Position-restricted substring searches;
- Other contributions of $\mathtt{rank}'$, $\mathtt{rmq}/\mathtt{rMq}$, and $\mathtt{succ}/\mathtt{pred}$ queries may be of general interest.

*Thank you!*