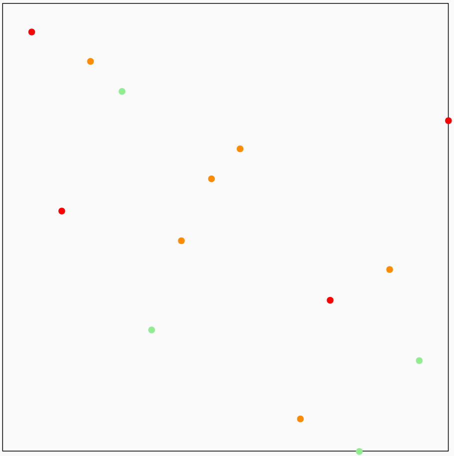
Space Efficient Two-Dimensional Orthogonal Colored Range Counting

Younan Gao and Meng He

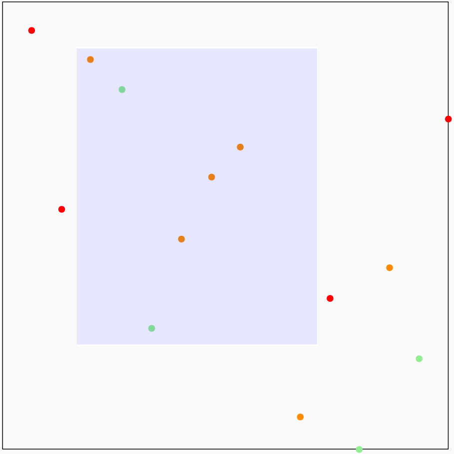
Dalhousie University

The Problem



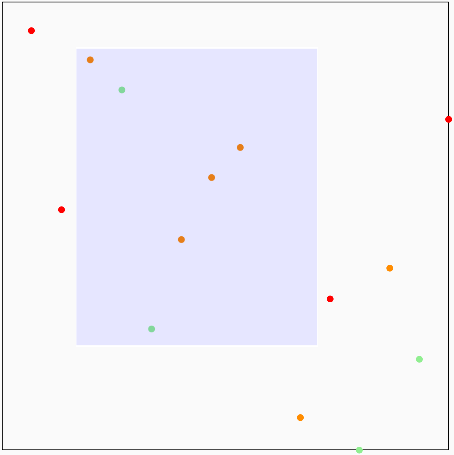
- The input is a set P of n points on the grid. Each point is assigned in some color, which is encoded by an integer $\in [1, C]$.

The Problem



- Given an orthogonal query range Q, compute the number (denoted by $|C(Q \cap P)|$) of distinct colors in $Q \cap P$, which is 2 in this example.

The Problem



- **Application in Database System**: SELECT COUNT (DISTINCT country) FROM athletes WHERE $a \leq$ weight $\leq b$ AND $c \leq$ height $\leq d$;

Related Work

	Space	Query Time
Non-Color Version	$O(n)$	$O(\frac{\lg n}{\lg\lg n})$
Color Version	$O(n^2\frac{\lg n}{\lg\lg n})$	$O((\frac{\lg n}{\lg\lg n})^2)$

Related Work

	Space	Query Time
Non-Color Version	$O(n)$	$O(\frac{\lg n}{\lg \lg n})$
Color Version	$O(n^2 \frac{\lg n}{\lg \lg n})$	$O((\frac{\lg n}{\lg \lg n})^2)$

Why is it hard?

Related Work

	Space	Query Time
Non-Color Version	$O(n)$	$O(\frac{\lg n}{\lg \lg n})$
Color Version	$O(n^2 \frac{\lg n}{\lg \lg n})$	$O((\frac{\lg n}{\lg \lg n})^2)$

Why is it hard?

- Not decomposable:

  Given $|C(P) \cap [a, b] \times [c, +\infty)|$ and $|C(P) \cap [a, b] \times (-\infty, d]|$,

  $|C(P) \cap [a, b] \times [c, d]|$ cannot be computed in constant time;

Related Work

	Space	Query Time
Non-Color Version	$O(n)$	$O(\frac{\lg n}{\lg\lg n})$
Color Version	$O(n^2\frac{\lg n}{\lg\lg n})$	$O((\frac{\lg n}{\lg\lg n})^2)$

Why is it hard?

- Not decomposable:
  Given $\lvert C(P)\cap[a,b]\times[c,+\infty)\rvert$ and $\lvert C(P)\cap[a,b]\times(-\infty,d]\rvert$,
  $\lvert C(P)\cap[a,b]\times[c,d]\rvert$ cannot be computed in constant time;
- Reduce 2D colored counting to Boolean Matrix Multiplication:
  No solution can simultaneously have preprocessing time better than $\Omega(n^{3/2})$ and query time better than $\Omega(\sqrt{n})$, by purely combinatorial methods

Related Work

	Space	Query Time
Non-Color Version	$O(n)$	$O(\frac{\lg n}{\lg \lg n})$
Color Version	$O(n^2 \frac{\lg n}{\lg \lg n})$	$O((\frac{\lg n}{\lg \lg n})^2)$

Why is it hard?

- Not decomposable:
  Given $|C(P) \cap [a, b] \times [c, +\infty)|$ and $|C(P) \cap [a, b] \times (-\infty, d]|$, $|C(P) \cap [a, b] \times [c, d]|$ cannot be computed in constant time;
- Reduce 2D colored counting to Boolean Matrix Multiplication:
  No solution can simultaneously have preprocessing time better than $\Omega(n^{3/2})$ and query time better than $\Omega(\sqrt{n})$, by purely combinatorial methods
- A solution with $O(n \lg^4 n)$ words and $O(\sqrt{n} \lg^8 n)$ query time by Kaplan et al 2008.

	Model	Query Time	Space Usage in Words
Kaplan et al.	PM	$O(X \lg^7 n)$	$O((\frac{n}{X})^2 \lg^6 n + n \lg^4 n)$
Sol.1	PM	$O(\lg^5 n + X \lg^3 n)$	$O((\frac{n}{X})^2 \lg^4 n + n \lg^3 n)$
	RAM	$O(\lg^4 n + X \lg^2 n \lg\lg n)$	$O((\frac{n}{X})^2 \lg^4 n + n \lg^3 n)$
Sol.2	RAM	$O(\lg^6 n + X \lg^{3+\epsilon} n)$	$O((\frac{n}{X})^2 \lg^4 n + n \lg^2 n)$
Sol.3	RAM	$O(\lambda^2 \lg^6 n \log^2_\lambda n + X \lg^{3+\epsilon} n \lambda \log_\lambda n)$	$O((\frac{n}{X})^2 \lg^2 n \log^2_\lambda n + n \lg n \log_\lambda n)$

	Model	Query Time	Space Usage in Words
Kaplan et al.	PM	$O(X \lg^7 n)$	$O((\frac{n}{X})^2 \lg^6 n + n \lg^4 n)$
Sol.1	PM	$O(\lg^5 n + X \lg^3 n)$	$O((\frac{n}{X})^2 \lg^4 n + n \lg^3 n)$
	RAM	$O(\lg^4 n + X \lg^2 n \lg\lg n)$	$O((\frac{n}{X})^2 \lg^4 n + n \lg^3 n)$
Sol.2	RAM	$O(\lg^6 n + X \lg^{3+\epsilon} n)$	$O((\frac{n}{X})^2 \lg^4 n + n \lg^2 n)$
Sol.3	RAM	$O(\lambda^2 \lg^6 n \log_\lambda^2 n + X \lg^{3+\epsilon} n \lambda \log_\lambda n)$	$O((\frac{n}{X})^2 \lg^2 n \log_\lambda^2 n + n \lg n \log_\lambda n)$
Kaplan et al.	PM	$O(\sqrt{n} \lg^8 n)$	$O(n \lg^4 n)$
Sol.1	PM	$O(\sqrt{n} \lg^{7/2} n)$	$O(n \lg^3 n)$
	RAM	$O(\sqrt{n} \lg^{5/2} n \lg\lg n)$	$O(n \lg^3 n)$
Sol.2	RAM	$O(\sqrt{n} \lg^{4+\epsilon} n)$	$O(n \lg^2 n)$
Sol.3	RAM	$O(\sqrt{n} \lg^{5+\epsilon} n)$	$O(n \frac{\lg^2 n}{\lg\lg n})$
	RAM	$O(n^{1/2+\epsilon})$	$O(n \lg n)$

	Model	Query Time	Space Usage in Words
Kaplan et al.	PM	$O(X \lg^7 n)$	$O((\frac{n}{X})^2 \lg^6 n + n \lg^4 n)$
Sol.1	PM	$O(\lg^5 n + X \lg^3 n)$	$O((\frac{n}{X})^2 \lg^4 n + n \lg^3 n)$
	RAM	$O(\lg^4 n + X \lg^2 n \lg \lg n)$	$O((\frac{n}{X})^2 \lg^4 n + n \lg^3 n)$
Sol.2	RAM	$O(\lg^6 n + X \lg^{3+\epsilon} n)$	$O((\frac{n}{X})^2 \lg^4 n + n \lg^2 n)$
Sol.3	RAM	$O(\lambda^2 \lg^6 n \log_\lambda^2 n + X \lg^{3+\epsilon} n \lambda \log_\lambda n)$	$O((\frac{n}{X})^2 \lg^2 n \log_\lambda^2 n + n \lg n \log_\lambda n)$
Kaplan et al.	PM	$O(\sqrt{n} \lg^8 n)$	$O(n \lg^4 n)$
Sol.1	PM	$O(\sqrt{n} \lg^{7/2} n)$	$O(n \lg^3 n)$
	RAM	$O(\sqrt{n} \lg^{5/2} n \lg \lg n)$	$O(n \lg^3 n)$
Sol.2	RAM	$O(\sqrt{n} \lg^{4+\epsilon} n)$	$O(n \lg^2 n)$
Sol.3	RAM	$O(\sqrt{n} \lg^{5+\epsilon} n)$	$O(n \frac{\lg^2 n}{\lg \lg n})$
	RAM	$O(n^{1/2+\epsilon})$	$O(n \lg n)$
Grossi and Vind	RAM	$O(n / \operatorname{polylog}(n))$	$O(n)$
Kaplan et al.	RAM	$O(n^{3/4} \lg^\epsilon n)$	$O(n)$

Overview of the Techniques

- Techniques:
    - Decomposing a 4-sided query range to two 3-sided subranges with a range tree;
    - Achieving new time-space tradeoffs when computing the number of colors that exist in both subranges. (Main contribution)

Preliminaries

2D 3-Sided Colored Range Counting

$\downarrow$

3D 3-Sided Colored Range Counting

Preliminaries

2D 3-Sided Colored Range Counting

$\downarrow$

3D 3-Sided Colored Range Counting

$\downarrow$

3D Stabbing Queries over 3D 5-Sided Boxes

Preliminaries

2D 3-Sided Colored Range Counting

$\downarrow$

3D 3-Sided Colored Range Counting

$\downarrow$ ✓

3D Stabbing Queries over 3D 5-Sided Boxes

Preliminaries

2D 3-Sided Colored Range Counting

$\downarrow$

3D 3-Sided Colored Range Counting

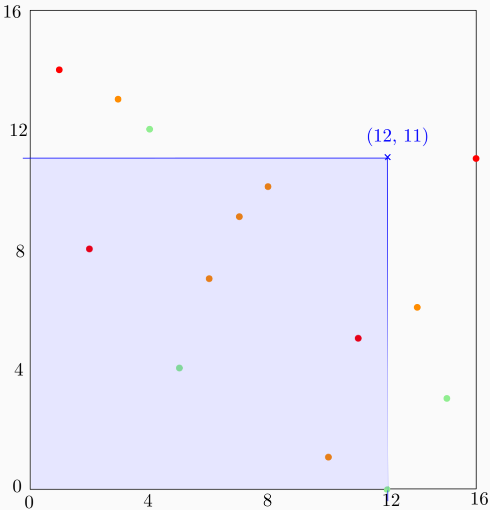
$\downarrow$ ✓

3D Stabbing Queries over 3D 5-Sided Boxes ✓

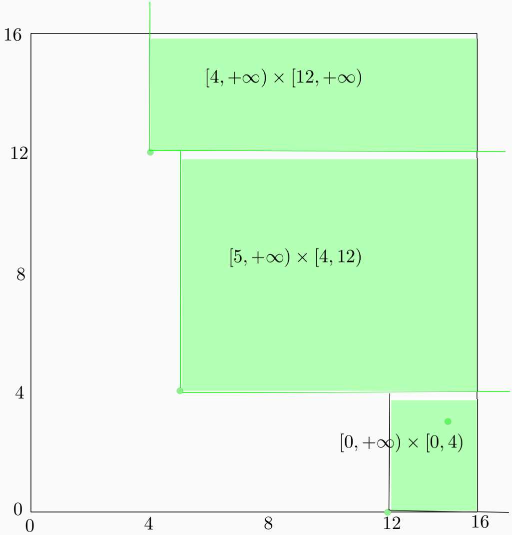
3D Dominance Colored Counting to 3D Stabbing Queries

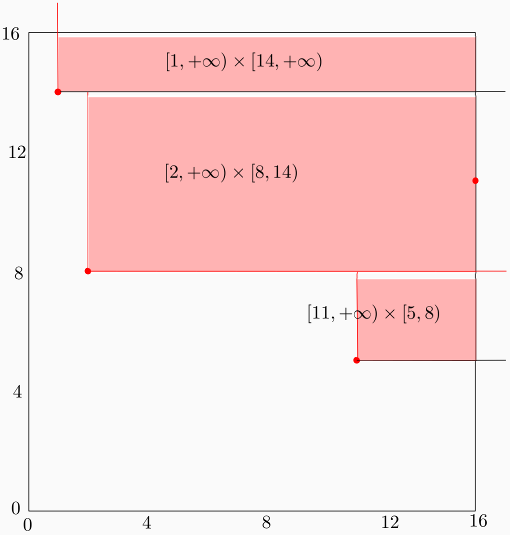
Lemma (Kaplan et al.)

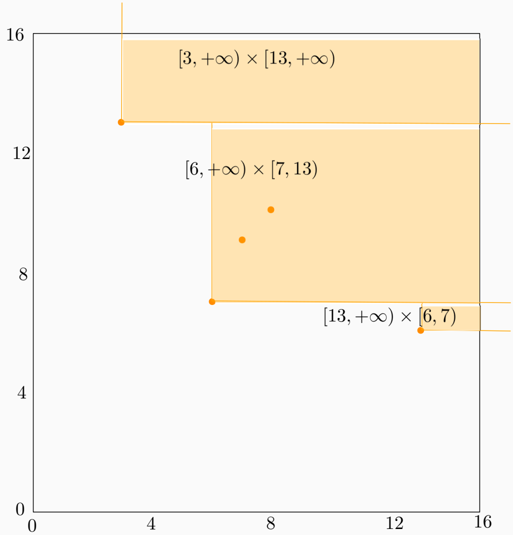
Given a set, P, of n points in three-dimensional space, we can assign points of P into a set $B(P)$ of $O(n)$ pairwise disjoint 3D canonical boxes (i.e., each in the form of $[x_1, +\infty) \times [y_1, y_2) \times [z_1, z_2)$) such that a query point q dominates some point in P iff q is contained in exactly one of the boxes from $B(P)$.



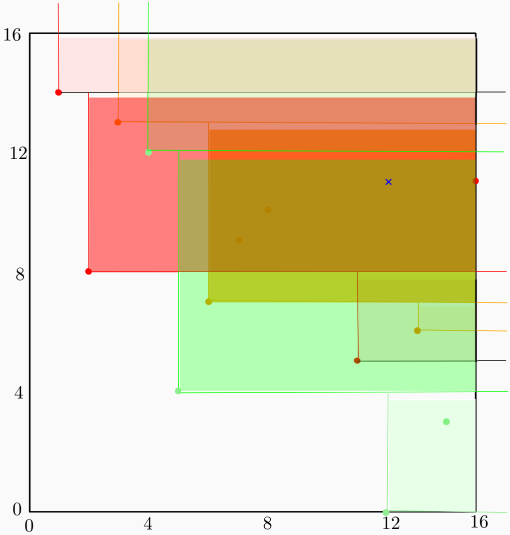
16
12
8
4
0
0
4
8
12
16
(12, 11)


$[4, +\infty) \times [12, +\infty)$
$[5, +\infty) \times [4, 12)$
$[0, +\infty) \times [0, 4)$
16
12
8
4
0
0
4
8
12
16


16
12
8
4
0
0
4
8
12
16
$[1, +\infty) \times [14, +\infty)$
$[2, +\infty) \times [8, 14)$
$[11, +\infty) \times [5, 8)$



16
12
8
4
0
0
4
8
12
16
$[3, +\infty) \times [13, +\infty)$
$[6, +\infty) \times [7, 13)$
$[13, +\infty) \times [6, 7)$


16
12
8
4
0
0
4
8
12
16

- Stabbing Queries is to report/count the number of canonical boxes containing the query point q;

- Stabbing Queries is to report/count the number of canonical boxes containing the query point q;
- In total $O(n)$ boxes will be built over all colors;

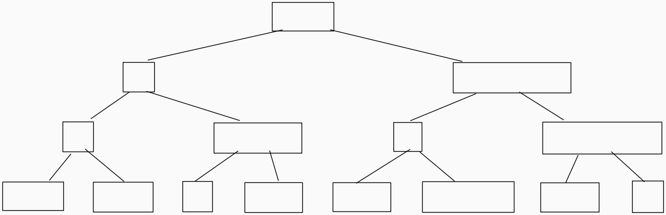
- Stabbing Queries is to report/count the number of canonical boxes containing the query point q;
- In total $O(n)$ boxes will be built over all colors;
- Within each color, at most one box contains the query point;

- Stabbing Queries is to report/count the number of canonical boxes containing the query point q;
- In total $O(n)$ boxes will be built over all colors;
- Within each color, at most one box contains the query point;
- The number of colors of the points dominated by q is the same as the number of boxes containing q;

- Stabbing Queries is to report/count the number of canonical boxes containing the query point q;
- In total $O(n)$ boxes will be built over all colors;
- Within each color, at most one box contains the query point;
- The number of colors of the points dominated by q is the same as the number of boxes containing q;

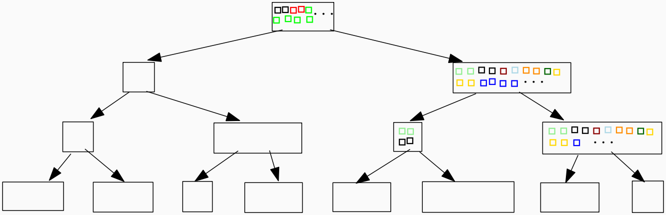
Stabbing Queries over 3D 5-Sided Boxes

- It is a two-layer segment tree structure;



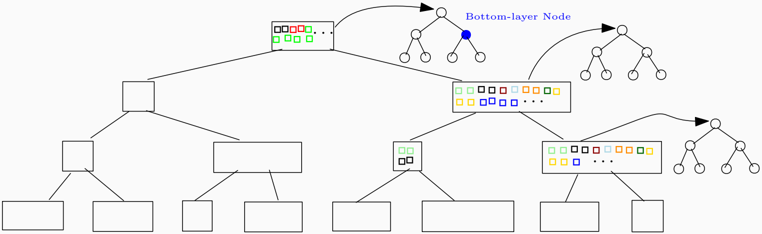
Stabbing Queries over 3D 5-Sided Boxes

- It is a two-layer segment tree structure;
- The first-layer is contructed upon the intervals of the boxes along z-axis;



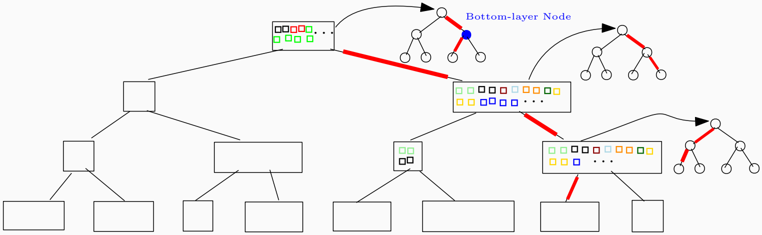
Stabbing Queries over 3D 5-Sided Boxes

- It is a two-layer segment tree structure;
- The first-layer is contructed upon the intervals of the boxes along z-axis;
- The second-layer is contructed upon the intervals of the boxes along y-axis;



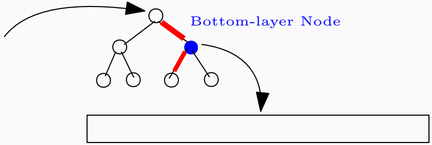
Stabbing Queries over 3D 5-Sided Boxes

- It is a two-layer segment tree structure;
- The first-layer is contructed upon the intervals of the boxes along z-axis;
- The second-layer is contructed upon the intervals of the boxes along y-axis;
- A query accesses $O(\lg^2 n)$ bottom-layer node;



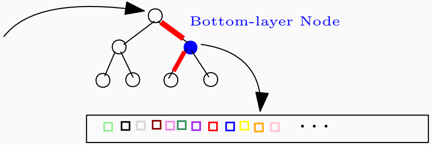
Stabbing Queries over 3D 5-Sided Boxes

- Boxes stored in the bottom-layer node are sorted by one-side bounded x-coordinate;



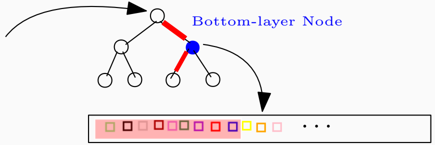
Stabbing Queries over 3D 5-Sided Boxes

- Boxes stored in the bottom-layer node are sorted by one-side bounded x-coordinate;
- Boxes stored in the same bottom-layer node are in distinct color;



Stabbing Queries over 3D 5-Sided Boxes

- Boxes stored in the bottom-layer node are sorted by one-side bounded x-coordinate;
- Boxes stored in the same bottom-layer node are in distinct color;
- A query upon a bottom-layer node would return some prefix of the box list (as reporting) or the size of the prefix list (as counting);



Stabbing Queries over 3D 5-Sided Boxes

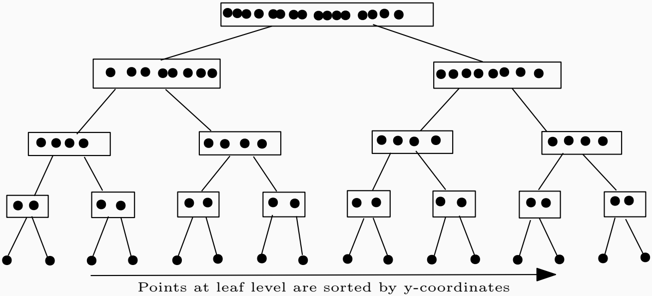
- The data structure uses $O(n \lg^2 n)$ words of space;

Stabbing Queries over 3D 5-Sided Boxes

- The data structure uses $O(n \lg^2 n)$ words of space;
- Each counting/reporting query takes $O(\lg^2 n)/O(\lg^2 n + k)$ time;

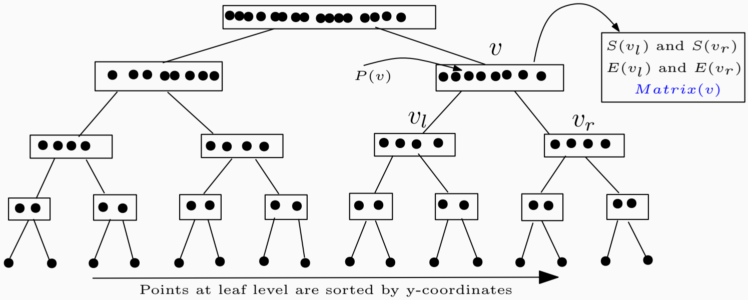
The First Solution

- Construct a binary range tree T upon y-coordinates of points.



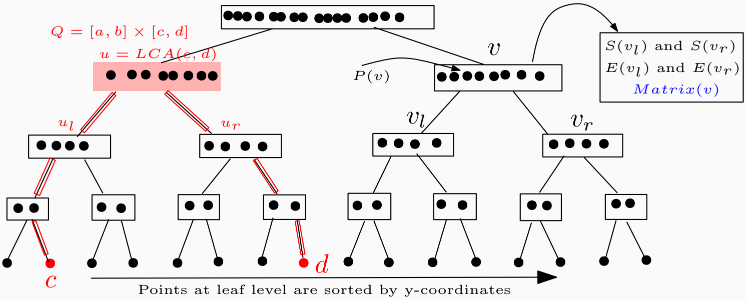
The First Solution

- Construct a binary range tree T upon y-coordinates of points.
- Construct and store $S(v_l)$ and $S(v_r)$ for 2D 3-sided colored counting upon $P(v_l)$ and $P(v_r)$ at node v.
- Construct and store $E(v_l)$ and $E(v_r)$ for 2D colored emptiness queries upon $P(v_l)$ and $P(v_r)$ at node v.



The First Solution

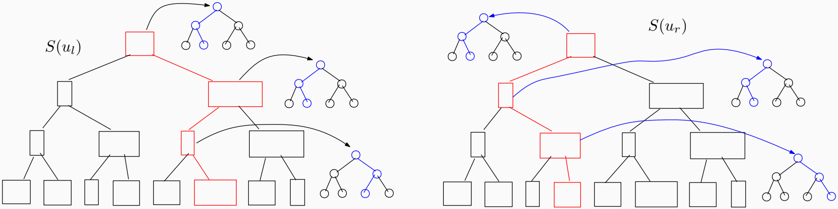
- Construct a binary range tree T upon y-coordinates of points.
- Construct and store $S(v_l)$ and $S(v_r)$ for 2D 3-sided colored counting upon $P(v_l)$ and $P(v_r)$ at node v.
- Construct and store $E(v_l)$ and $E(v_r)$ for 2D colored emptiness queries upon $P(v_l)$ and $P(v_r)$ at node v.



- By the exclusion-inclusion principle, we know that $|C_Q(u)| = |C_Q(u_l)| + |C_Q(u_r)| - |C_Q(u_l) \cap C_Q(u_r)|$.

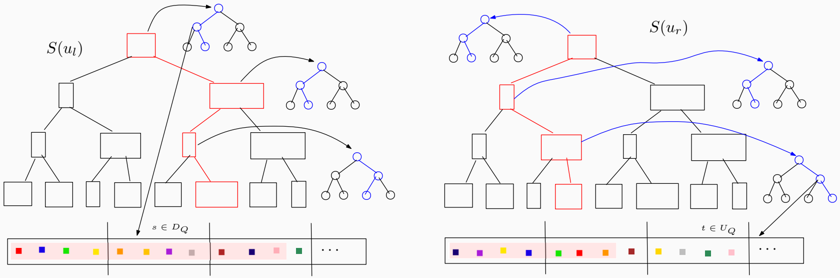
Computing the Intersection

- We denote D_Q and U_Q to be the sets of the $O(\lg^2 n)$ prefix lists on the accessed bottom-layer nodes of $S(u_l)$ and $S(u_r)$, respectively.



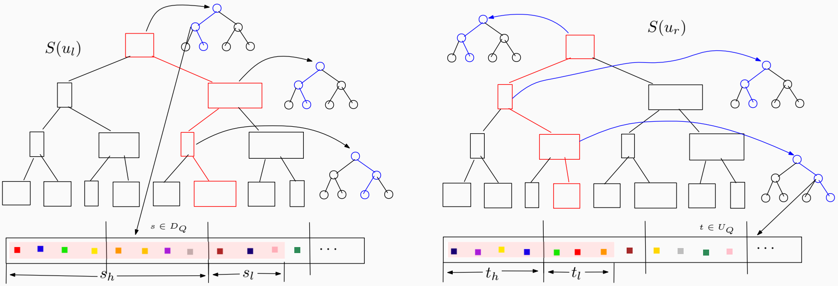
Computing the Intersection

- We denote D_Q and U_Q to be the sets of the $O(\lg^2 n)$ prefix lists on the accessed bottom-layer nodes of $S(u_l)$ and $S(u_r)$, respectively.
- In the preprocessing, each list stored at the bottom-layer node is divided into blocks of size X;



Computing the Intersection

- We denote D_Q and U_Q to be the sets of the $O(\lg^2 n)$ prefix lists on the accessed bottom-layer nodes of $S(u_l)$ and $S(u_r)$, respectively.
- In the preprocessing, each list stored at the bottom-layer node is divided into blocks of size X;



- We have $C(s_h) \cup C(s_l) = C(s)$ and $C(t_h) \cup C(t_l) = C(t)$; and $C(s_h) \cap C(s_l) = C(t_h) \cap C(t_l) = \emptyset$ also holds;

$|C_Q(u_l) \cap C_Q(u_r)|$ can be computed in $O(\lg^4 n + X \lg^2 n \lg\lg n)$ time.

$$|C_Q(u_l) \cap C_Q(u_r)| = \sum_{s \in D_Q, t \in U_Q} |C(s) \cap C(t)|$$

$|C_Q(u_l) \cap C_Q(u_r)|$ can be computed in $O(\lg^4 n + X \lg^2 n \lg\lg n)$ time.

$$|C_Q(u_l) \cap C_Q(u_r)| = \sum_{s \in D_Q, t \in U_Q} |C(s) \cap C(t)|$$

$$= \sum_{s \in D_Q, t \in U_Q} (|(C(s_h) \cup C(s_l)) \cap (C(t_h) \cup C(t_l))|$$

$|C_Q(u_l) \cap C_Q(u_r)|$ can be computed in $O(\lg^4 n + X \lg^2 n \lg\lg n)$ time.

$$|C_Q(u_l) \cap C_Q(u_r)| = \sum_{s \in D_Q, t \in U_Q} |C(s) \cap C(t)|$$

$$= \sum_{s \in D_Q, t \in U_Q} (|(C(s_h) \cup C(s_l)) \cap (C(t_h) \cup C(t_l))|$$

$$= \sum_{s \in D_Q, t \in U_Q} (|(C(s_h) \cap C(t_h)) \cup (C(s_h) \cap C(t_l)) \cup (C(s_l) \cap C(t)|))$$

$|C_Q(u_l) \cap C_Q(u_r)|$ can be computed in $O(\lg^4 n + X \lg^2 n \lg\lg n)$ time.

$$
\begin{aligned}
&|C_Q(u_l) \cap C_Q(u_r)| = \sum_{s \in D_Q, t \in U_Q} |C(s) \cap C(t)| \\
&= \sum_{s \in D_Q, t \in U_Q} (|(C(s_h) \cup C(s_l)) \cap (C(t_h) \cup C(t_l))| \\
&= \sum_{s \in D_Q, t \in U_Q} (|(C(s_h) \cap C(t_h)) \cup (C(s_h) \cap C(t_l)) \cup (C(s_l) \cap C(t)|)) \\
&= \sum_{s \in D_Q, t \in U_Q} (|C(s_h) \cap C(t_h)| + |C(s_h) \cap C(t_l)| + |C(s_l) \cap C(t)|)
\end{aligned}
$$

- $\sum_{s \in D_Q, t \in U_Q} (|C(s_h) \cap C(t_h)|$ can be computed in $O(\lg^2(n) \times \lg^2(n)) = O(\lg^4 n)$ time;

$|C_Q(u_l) \cap C_Q(u_r)|$ can be computed in $O(\lg^4 n + X \lg^2 n \lg\lg n)$ time.

$$|C_Q(u_l) \cap C_Q(u_r)| = \sum_{s \in D_Q, t \in U_Q} |C(s) \cap C(t)|$$

$$= \sum_{s \in D_Q, t \in U_Q} (|(C(s_h) \cup C(s_l)) \cap (C(t_h) \cup C(t_l))|$$

$$= \sum_{s \in D_Q, t \in U_Q} (|(C(s_h) \cap C(t_h)) \cup (C(s_h) \cap C(t_l)) \cup (C(s_l) \cap C(t)|))$$

$$= \sum_{s \in D_Q, t \in U_Q} (|C(s_h) \cap C(t_h)| + |C(s_h) \cap C(t_l)| + |C(s_l) \cap C(t)|)$$

- $\sum_{s \in D_Q, t \in U_Q} |C(s_l) \cap C(t)| = |(\cup_{s \in D_Q} C(s_l)) \cap C_Q(u_r)|$ can be computed in $O(X \times \lg^2 n \times \lg\lg n) = O(X \lg^2 n \lg\lg n)$ time;

$|C_Q(u_l) \cap C_Q(u_r)|$ **can be computed in** $O(\lg^4 n + X \lg^2 n \lg\lg n)$ **time.**

$$\sum_{s \in D_Q, t \in U_Q} |C(s_h) \cap C(t_l)|$$

$|C_Q(u_l) \cap C_Q(u_r)|$ **can be computed in** $O(\lg^4 n + X \lg^2 n \lg\lg n)$ **time.**

$$
\begin{aligned}
&\sum_{s \in D_Q, t \in U_Q} |C(s_h) \cap C(t_l)| \\
= &\sum_{s \in D_Q, t \in U_Q} |(C(s)/C(s_l)) \cap C(t_l)|
\end{aligned}
$$

$|C_Q(u_l) \cap C_Q(u_r)|$ can be computed in $O(\lg^4 n + X \lg^2 n \lg\lg n)$ time.

$$
\begin{aligned}
&\sum_{s \in D_Q, t \in U_Q} |C(s_h) \cap C(t_l)| \\
=&\sum_{s \in D_Q, t \in U_Q} |(C(s)/C(s_l)) \cap C(t_l)| \\
=&|(\cup_{s \in D_Q} C(s)) \cap (\cup_{t \in U_Q} C(t_l))| - |(\cup_{s \in D_Q} C(s_l)) \cap (\cup_{t \in U_Q} C(t_l))|
\end{aligned}
$$

$|C_Q(u_l) \cap C_Q(u_r)|$ can be computed in $O(\lg^4 n + X \lg^2 n \lg\lg n)$ time.

$$
\begin{aligned}
& \sum_{s \in D_Q, t \in U_Q} |C(s_h) \cap C(t_l)| \\
= & \sum_{s \in D_Q, t \in U_Q} |(C(s)/C(s_l)) \cap C(t_l)| \\
= & |(\cup_{s \in D_Q} C(s)) \cap (\cup_{t \in U_Q} C(t_l))| - |(\cup_{s \in D_Q} C(s_l)) \cap (\cup_{t \in U_Q} C(t_l))| \\
= & |C_Q(u_l) \cap (\cup_{t \in U_Q} C(t_l))| - |(\cup_{s \in D_Q} C(s_l)) \cap (\cup_{t \in U_Q} C(t_l))|
\end{aligned}
$$

$|C_Q(u_l) \cap C_Q(u_r)|$ can be computed in $O(\lg^4 n + X \lg^2 n \lg\lg n)$ time.

$$
\begin{aligned}
&\sum_{s \in D_Q, t \in U_Q} |C(s_h) \cap C(t_l)| \\
=&\sum_{s \in D_Q, t \in U_Q} |(C(s)/C(s_l)) \cap C(t_l)| \\
=&|(\cup_{s \in D_Q} C(s)) \cap (\cup_{t \in U_Q} C(t_l))| - |(\cup_{s \in D_Q} C(s_l)) \cap (\cup_{t \in U_Q} C(t_l))| \\
=&|C_Q(u_l) \cap (\cup_{t \in U_Q} C(t_l))| - |(\cup_{s \in D_Q} C(s_l)) \cap (\cup_{t \in U_Q} C(t_l))| \\
=&|C_Q(u_l) \cap (\cup_{t \in U_Q} C(t_l))| - |(\cup_{s \in D_Q} C(s_l)) \cap (\cup_{t \in U_Q} C(t_l))|
\end{aligned}
$$

- $|C_Q(u_l) \cap (\cup_{t \in U_Q} C(t_l))|$ can be computed in $O(X \times \lg^2 n \times \lg\lg n) = O(X \lg^2 n \cdot \lg\lg n)$ time;

$|C_Q(u_l) \cap C_Q(u_r)|$ can be computed in $O(\lg^4 n + X \lg^2 n \lg\lg n)$ time.

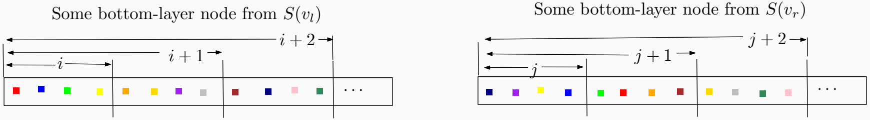
$$
\begin{aligned}
&\sum_{s \in D_Q, t \in U_Q} |C(s_h) \cap C(t_l)| \\
= &\sum_{s \in D_Q, t \in U_Q} |(C(s)/C(s_l)) \cap C(t_l)| \\
= &|(\cup_{s \in D_Q} C(s)) \cap (\cup_{t \in U_Q} C(t_l))| - |(\cup_{s \in D_Q} C(s_l)) \cap (\cup_{t \in U_Q} C(t_l))| \\
= &|C_Q(u_l) \cap (\cup_{t \in U_Q} C(t_l))| - |(\cup_{s \in D_Q} C(s_l)) \cap (\cup_{t \in U_Q} C(t_l))|
\end{aligned}
$$

$$
=|C_Q(u_l) \cap (\cup_{t \in U_Q} C(t_l))| - |(\cup_{s \in D_Q} C(s_l)) \cap (\cup_{t \in U_Q} C(t_l))|
$$

- Since colors are encoded in integers, $|(\cup_{s \in D_Q} C(s_l)) \cap (\cup_{t \in U_Q} C(t_l))|$ can be computed in $O(X \times \lg^2 n \times \lg\lg n) = O(X \lg^2 n \cdot \lg\lg n)$ time by sorting;

Matrixes for storing all pairs of $|s_h \cap t_h|$

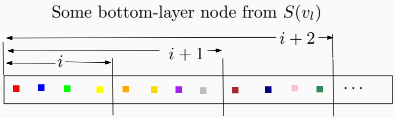
- We store a matrix $M(v)$ at each internal node v of T.

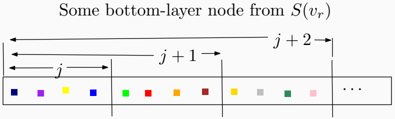


M(v)	$\cdots$	j	j+1	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
i	$\cdots$	2	4	$\cdots$
i+1	$\cdots$	3	6	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$

Matrixes for storing all pairs of $|s_h \cap t_h|$

- We store a matrix $M(v)$ at each internal node v of T.
- Within $M(v)$, there are $O(\frac{|P(v_l)|\lg^2 n}{X})$ rows and $O(\frac{|P(v_r)|\lg^2 n}{X})$ columns.



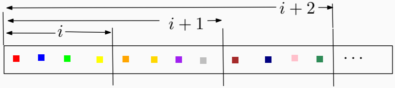


M(v)	1	$\cdots$	j	j+1	$\cdots$	$O(\frac{\lvert P(v_r)\rvert\lg^2 n}{X})$
1	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
i	$\cdots$	$\cdots$	2	4	$\cdots$	$\cdots$
i+1	$\cdots$	$\cdots$	3	6	$\cdots$	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$O(\frac{\lvert P(v_l)\rvert\lg^2 n}{X})$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$

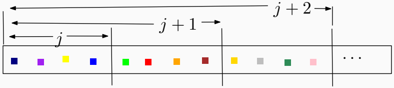
Matrixes for storing all pairs of $|s_h \cap t_h|$

- We store a matrix $M(v)$ at each internal node v of T.
- Within $M(v)$, there are $O(\frac{|P(v_l)| \lg^2 n}{X})$ rows and $O(\frac{|P(v_r)| \lg^2 n}{X})$ columns.
- $M(v)$ uses $O((\frac{|P(v)| \lg^2 n}{X})^2)$ words of space.

Some bottom-layer node from $S(v_l)$



Some bottom-layer node from $S(v_r)$



M(v)	1	$\cdots$	j	j+1	$\cdots$	$O(\frac{\lvert P(v_r)\rvert \lg^2 n}{X})$
1	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
i	$\cdots$	$\cdots$	2	4	$\cdots$	$\cdots$
i+1	$\cdots$	$\cdots$	3	6	$\cdots$	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$O(\frac{\lvert P(v_l)\rvert \lg^2 n}{X})$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$

Summary of the First Solution

- All $E(v_l)$ and $E(v_r)$ at the same tree level use $O(n \lg \lg n)$ words of space.

Summary of the First Solution

- All $E(v_l)$ and $E(v_r)$ at the same tree level use $O(n \lg \lg n)$ words of space.
- All $S(v_l)$ and $S(v_r)$ at the same tree level use $O(n \lg^2 n)$ words of space.

Summary of the First Solution

- All $E(v_l)$ and $E(v_r)$ at the same tree level use $O(n \lg \lg n)$ words of space.
- All $S(v_l)$ and $S(v_r)$ at the same tree level use $O(n \lg^2 n)$ words of space.
- All $M(v)$ at the same tree level use $O((\frac{n \lg^2 n}{X})^2)$ words of space.

Summary of the First Solution

- All $E(v_l)$ and $E(v_r)$ at the same tree level use $O(n \lg \lg n)$ words of space.
- All $S(v_l)$ and $S(v_r)$ at the same tree level use $O(n \lg^2 n)$ words of space.
- All $M(v)$ at the same tree level use $O((\frac{n \lg^2 n}{X})^2)$ words of space.
- The binary range tree T has $O(\lg n)$ tree levels.

Summary of the First Solution

- All $E(v_l)$ and $E(v_r)$ at the same tree level use $O(n \lg \lg n)$ words of space.
- All $S(v_l)$ and $S(v_r)$ at the same tree level use $O(n \lg^2 n)$ words of space.
- All $M(v)$ at the same tree level use $O((\frac{n \lg^2 n}{X})^2)$ words of space.
- The binary range tree T has $O(\lg n)$ tree levels.
- The space cost is $O((\frac{n}{X})^2 \lg^4 n + n \lg^3 n)$ words, and its query time is $O(\lg^4 n + X \lg^2 n \lg \lg n)$.

The Second Method

Stabbing queries over 3D 5-Sided Boxes

The Second Method

Stabbing queries over 3D 5-Sided Boxes

$$\Bigg\downarrow \text{Segment Tree}$$

Stabbing queries over 2D 3-Sided Boxes

The Second Method

Stabbing queries over 3D 5-Sided Boxes

$\Big\downarrow$ Segment Tree

Stabbing queries over 2D 3-Sided Boxes

$\Big\downarrow$ Interval Tree

2D Dominance Range Searching

The Second Method

Stabbing queries over 3D 5-Sided Boxes

$\downarrow$ Segment Tree

Stabbing queries over 2D 3-Sided Boxes

$\downarrow$ Interval Tree

2D Dominance Range Searching

- The space cost of 3D stabbing query structure is $O(n \lg n)$ words;

The Second Method

$$\begin{array}{c}\text{Stabbing queries over 3D 5-Sided Boxes}\\ \Big\downarrow{\scriptstyle\text{Segment Tree}}\\ \text{Stabbing queries over 2D 3-Sided Boxes}\\ \Big\downarrow{\scriptstyle\text{Interval Tree}}\\ \text{2D Dominance Range Searching}\end{array}$$

- The space cost of 3D stabbing query structure is $O(n \lg n)$ words;
- The second solution uses $O((\frac{n}{X})^2 \lg^4 n + n \lg^2 n)$ words of space and achieves $O(\lg^6 n + X \lg^{3+\epsilon} n)$ query time for any constant $\epsilon \in (0, 1)$;

The Second Method

Stabbing queries over 3D 5-Sided Boxes

$\Big\downarrow$ Segment Tree

Stabbing queries over 2D 3-Sided Boxes

$\Big\downarrow$ Interval Tree

2D Dominance Range Searching

- The space cost of 3D stabbing query structure is $O(n \lg n)$ words;
- The second solution uses $O((\frac{n}{X})^2 \lg^4 n + n \lg^2 n)$ words of space and achieves $O(\lg^6 n + X \lg^{3+\epsilon} n)$ query time for any constant $\epsilon \in (0, 1)$;
- Setting $X = \sqrt{n} \lg n$ achieves $O(n \lg^2 n)$ space and $O(\sqrt{n} \lg^{4+\epsilon} n)$ query time;

The Third Method

Stabbing queries over 3D 5-Sided Boxes

The Third Method

Stabbing queries over 3D 5-Sided Boxes

$\Big\downarrow$ Interval Tree

Stabbing queries over 3D 4-Sided Boxes

The Third Method

Stabbing queries over 3D 5-Sided Boxes

$\downarrow$ Interval Tree

Stabbing queries over 3D 4-Sided Boxes

$\downarrow$ Interval Tree

3D Dominance Range Searching

The Third Method

Stabbing queries over 3D 5-Sided Boxes

$\downarrow$ Interval Tree

Stabbing queries over 3D 4-Sided Boxes

$\downarrow$ Interval Tree

3D Dominance Range Searching

- The space cost of 3D stabbing query structure is $O(n \log_\lambda n)$ words;

The Third Method

$$
\begin{array}{c}
\text{Stabbing queries over 3D 5-Sided Boxes} \\
\Big\downarrow \text{Interval Tree} \\
\text{Stabbing queries over 3D 4-Sided Boxes} \\
\Big\downarrow \text{Interval Tree} \\
\text{3D Dominance Range Searching}
\end{array}
$$

- The space cost of 3D stabbing query structure is $O(n \log_\lambda n)$ words;
- The third solution uses $O((\frac{n}{X})^2 \lg^2 n \cdot \log_\lambda^2 n + n \lg n \cdot \log_\lambda n)$ space and achieves $O(\lambda^2 \cdot \lg^6 n \cdot \log_\lambda^2 n + X \cdot \lg^{3+\epsilon} n \cdot \lambda \log_\lambda n)$ query time for an integer parameter $\lambda \in [2, n]$;

The Third Method

$$\begin{array}{c}\text{Stabbing queries over 3D 5-Sided Boxes}\\ \Big\downarrow \text{Interval Tree}\\ \text{Stabbing queries over 3D 4-Sided Boxes}\\ \Big\downarrow \text{Interval Tree}\\ \text{3D Dominance Range Searching}\end{array}$$

- The space cost of 3D stabbing query structure is $O(n \log_\lambda n)$ words;
- The third solution uses $O((\frac{n}{X})^2 \lg^2 n \cdot \log_\lambda^2 n + n \lg n \cdot \log_\lambda n)$ space and achieves $O(\lambda^2 \cdot \lg^6 n \cdot \log_\lambda^2 n + X \cdot \lg^{3+\epsilon} n \cdot \lambda \log_\lambda n)$ query time for an integer parameter $\lambda \in [2, n]$;
- Setting $X = \sqrt{n \lg n}$ and $\lambda = n^{\epsilon/5}$ achieves $O(n \lg n)$ space and $O(n^{1/2+\epsilon})$ query time.

Open Problem

- The most space-efficient solution we achieved uses $O(n \lg n)$ words of space and its query time is $O(n^{1/2+\epsilon})$;

Open Problem

- The most space-efficient solution we achieved uses $O(n \lg n)$ words of space and its query time is $O(n^{1/2+\epsilon})$;
- The most efficient query time using a linear space data strucure requires $O(n^{3/4} \lg^{\epsilon} n)$ time;

Open Problem

- The most space-efficient solution we achieved uses $O(n \lg n)$ words of space and its query time is $O(n^{1/2+\epsilon})$;
- The most efficient query time using a linear space data strucure requires $O(n^{3/4} \lg^{\epsilon} n)$ time;
- **Open Problem:** Can we build a linear space data structure supporting 2D colored range counting in $o(n^{3/4})$ time?

Questions?