Point Location in Sublog Time: Orthogonal Case

Assumption: word RAM model, \( w \approx \log U \)
integer input \( w \approx \log U \)

eqw.

Slab Method: \( O(\log \log U) \) time (by uEB tree)
\( O(n^2) \) space

de Berg-Snoeyink-van Kreveld '95:
\( O(n) \) space
\( O((\log \log U)^2) \) time

\( \rightarrow \)
C. 'Ill:
\( O(n) \) space
\( O(\log \log U) \) time

Approach 1

idea - use \( \sqrt{n} \times \sqrt{n} \) grid

let universe be \( [u_x] \times [u_y] \)
form \( \sqrt{n} \) columns of width \( \frac{u_x}{\sqrt{n}} \)
& \( \sqrt{n} \) rows of height \( \frac{u_y}{\sqrt{n}} \)

Given \( n \) disjoint rects:
1. for each column/row \( \sigma \),
   recursively build DS for all rects that have a vertex in \( \sigma \)

2. for each grid cell \( \tau \),
   record info (is \( \tau \) completely inside a rect. ?) (Note: each rect. is stored horizontally/vertically \( \leq 4 \) subrects)

Query alg'm, given pt \( q \):
find grid cell \( \tau \) containing \( q \) \( \leq O(1) \) time
if \( \tau \) is completely inside a rectangle
done
if \( \tau \) is cut across horizontally then recurse in \( \tau \)'s row
   " " " " " vertically " " " " " \( \tau \)'s cells
else recurse in \( \tau \)'s row or column
(Note: just 1 recursive call!)

\[ Q(n, U_x, U_y) \leq \begin{cases} 
Q(n, U_x, U_y) + O(1) & \text{or} \\
Q(n, U_x, \frac{U_y}{\sqrt{n}}) + O(1) 
\end{cases} \]

[good initially but not good as \( n \) gets small]
[could reduce to rank space to make it, \( W = O(n) \), but extra \( \log \log \) ]

**Approach 2**

- idea - generalize vEB
- divide into \( V \times x \) columns of width \( U_x \)

1. let \( D = \) set of all nonempty columns
2. recursively build DS after rounding

- universe size \( \sqrt{U_x} \times U_y \)

| "top structure!" |

3. recursively build DS for all nonempty columns, glued into one!

- universe size \( \leq n \sqrt{U_x} \times U_y \)

| "bottom structure" |

(Note: each rect. is stored in \( \leq 2 \) structs)
Query algorithm, given $q$:

- Find column $σ$ containing $q$ in $O(1)$ time.
- If $σ \notin D$ then recurse in top structure.
- Else recurse in bottom structure.

\[ Q(n, U_x, U_y) \leq Q(n, nVU_x, U_y) + O(1), \quad (***) \]

[ $U_x$ converges to $O(n)$, not const! ]

**Approach 3**

Same but $'n$ in $y$.

\[ Q(n, U_x, U_y) \leq Q(n, U_x, nVU_y) + O(1), \quad (***) \]

**Final Approach**

Combine!

**Case 1.** $n \geq U_x^{y_3}$ and $n \geq U_y^{y_3}$

\[(*) \Rightarrow Q(n, U_x, U_y) \leq \begin{cases} Q(n, U_x^{5/6}, U_y) + O(1) & \text{or} \\ Q(n, U_x, U_y^{5/6}) + O(1) \end{cases} \]

**Case 2.** $n < U_x^{y_3}$

\[(**) \Rightarrow Q(n, U_x, U_y) \leq Q(n, U_x^{5/6}, U_y) + O(1) \]

**Case 3.** $n < U_y^{y_3}$

\[(***) \Rightarrow Q(n, U_x, U_y) \leq Q(n, U_x, U_y^{5/6}) + O(1) \]

# Levels of recursion:

\[O(\log\log U_x + \log\log U_y)\]

\[\Rightarrow \text{query time } O(\log\log U)\]
Space $O(n \cdot 4^{\omega \log \log n})$ 

$= O(n (\log n)^{\alpha_n})$

can be reduced to $O(n)$ by separator method or sampling.