Point Location in Sublog Time: General Case
(Assumption: coords are integers in \( I \cup J \), \( w > \log U \), \( w > \log n \))

C. Patrascu '06:
\[ O(n) \text{ space, } O(\frac{\log n}{\log \log n}) \text{ time} \] or
\[ O\left(\frac{\sqrt{\log U}}{\log \log U}\right) \text{ time} \]

suffice to solve slab subproblem:
apply slab method, \( \Rightarrow \) \( O(n^3) \) space for general problem
then apply separator method to reduce space...

Method 1:

\[ \text{vEB trees don't work, can't hash...} \]
idea - \( b \)-way divide & conquer (like fusion tree)
- but can't pack \( b \) segs in a word
- sometimes reduce \( n \) sometimes value universe size
- \( \frac{U}{H} \)
- \( \frac{U^2}{H^2} \)
- \( \frac{U^3}{H^3} \)
- \( \frac{U^4}{H^4} \)

let \( U_L = \) "left universe size" \( U_R = \) "right universe size"

sort segs \( S \)
let \( B = \{ \text{segment } n/6, 2n/6, \ldots, \frac{U}{H} \} \)
\( (b-1) \times \frac{n}{b} \) in \( S \) \( \text{if } \abs{B} \leq b \)

divide left universe into subintervals of length \( \frac{U_L}{H} \)
\( S_1 = \) lowest seg
for \( i = 1, 3, S \)
\( S_i = \) highest seg in \( B \) that has left endpt in some subint \( \text{or right endpt of} \)

$s_{i+1} = \text{next seg in } B \text{ after } s_i$

$s_i$ = seg $s_i$ after rounding indices upward

Note: between $s_i$ & $s_{i+1}$,
- left universe size $\leq \frac{UL}{H}$
- or right universe size $\leq \frac{UR}{H}$

Q1: between $s_i$ & $s_{i+1}$
- # segs $\leq \frac{n}{b}$

3. $s_1 < s_2 < s_3 < s_4 < \ldots$

4. can pack $s_1$, $s_2$, $\ldots$, in a word
   assuming $\frac{b \log H}{H} \leq w$

Query alg', given pt $q$:
locate $q$ in $\{s_1, s_2, s_3, \ldots\}$ by word op in $O(1)$ time
with 1 extra comp, can locate $q$ in $\{s_1, s_2, s_3, \ldots\}$
with 1 more comp, can locate $q$ in $\{s_1, s_2, s_3, \ldots\}$
say $s_i < q < s_{i+1}$
recurse between $s_i$ & $s_{i+1}$

$Q(n, U_L, U_R) \leq \left\{ \begin{array}{ll}
Q(n, \frac{UL}{H}, \frac{UR}{H}) + O(1) & \text{or} \\
Q(n, U_L, \frac{UR}{H}) + O(1) & \text{or} \\
Q(\frac{n}{b}, U_L, U_R) + O(1) & \text{or}
\end{array} \right.$

$\Rightarrow O\left( \log \frac{U_L}{H} + \log \frac{UR}{H} + \log_b n \right)$

$= O\left( \frac{\log U}{\log H} + \frac{\log n}{\log b} \right)$

Set $b = \frac{w}{\log H}$
\[
= O\left( \frac{b + \log n}{\log b} \right)
\]
set \( b = \log n \)

\[
= O\left( \frac{\log n}{\log \log n} \right)
\]

Note \( \text{OP} \) can be replaced by standard GSPs given \( 2^k \) segs each with \( \log H \) bits, \( 0 \leq \log H \leq w \).

Say segment \( i \) is from \((0, a_i)\) to \((H, b_i)\).

Want to compare \( y \leq \frac{b_i - a_i}{H} x + a_i \)

I.e. \( H y \leq (b_i - a_i) x + a_i \)

\[
= \frac{b_2 - a_2}{b_1 - a_1} - x
\]

must

\[
= \frac{(b_1 - a_1)x + (b_2 - a_2)x}{H a_1 + H a_2}
\]

add \( H a_1, H a_2 \)

Compare against \( H y \).

Subproblem: Compare a given value against list of \( b \) values (each \( \log H \) bits)
compare 0101
with list 0001 0011 0110 0111
add 1000 1000 1000 1000
     1001 1011 1110 1111
     0101

and 0001 0001 0001 0001
     0101 0101 0101 0101

Subtract 0100 0110 1001 1010

bitwise-and 1000 1000 1000 1000

MSB (most significant bit) (can avoid by multiplexing: combine gives count)

Method 2
set b = n this time, don't pack
solve subproblem for \( \tilde{s}_1, \tilde{s}_2, \tilde{s}_3, \ldots \) by table look-up.
(table size \( O(H^2) \))

\[ \Rightarrow \text{space} \quad O(H^2n) \]

query \( O(\frac{\log U}{\log H} + \frac{\log n}{\log b}) \)

To reduce space, select every other \( H^2 \) steps
\[ \Rightarrow \text{space} \quad O(n) \]

query \( O(\frac{\log U}{\log H} + \log H^2) \) (binary search

\[ = O(V\log U) \] by setting \( \log H = V\log U \)
better still $O\left( \frac{\log U}{\log H} + \frac{\log H}{\log \log H} \right)$

by method 1

by setting $\log H = \log U \log \log U$

[Better? open]

[offline queries: time $2 O(\log \log \log n)$; better than $\log n$]