

$\theta > A_1$

BST Application

Range Query: 1D

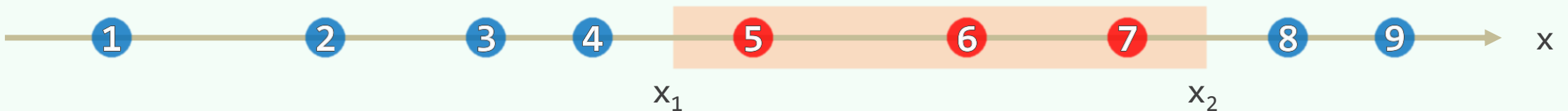
你这个人太敏感了。这个社会什么都需要，唯独不需要敏感。

邓俊辉

deng@tsinghua.edu.cn

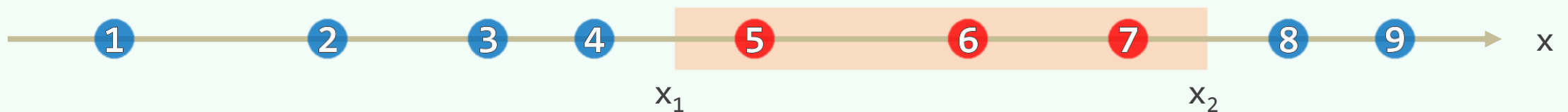
1D Range Query

- ❖ Let $P = \{ p_1, p_2, p_3, \dots, p_n \}$ be a set of n points on the x -axis
- ❖ For any given interval $I = (x_1, x_2]$
 - COUNTING: how many points of P lies in the interval?
 - REPORTING: enumerate all points in $I \cap P$ (if not empty)
- ❖ [Online] P is fixed while I is randomly and repeatedly given
- ❖ How to PREPROCESS P into a certain data structure s.t.
the queries can be answered efficiently?



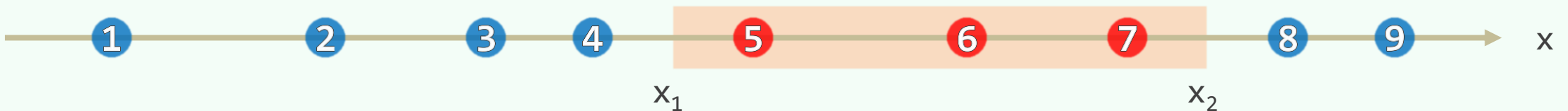
Brute-Force

- ❖ For each point p of P , test if $p \in (x_1, x_2]$
- ❖ Thus each query can be answered in LINEAR time
- ❖ Can we do it faster? It seems we can't, for ...
- ❖ In the worst case,
the interval contains up to $\mathcal{O}(n)$ points, which need $\mathcal{O}(n)$ time to enumerate
- ❖ However, how if we
ignore the time for **enumerating** and count only the **searching** time?



Binary Search

- ❖ Sort all points into a sorted vector and add an extra sentinel $p[0] = -\infty$
- ❖ For any interval $I = (x_1, x_2]$
 - Find $t = \text{search}(x_2) = \max\{ i \mid p[i] \leq x_2 \}$ $// \mathcal{O}(\log n)$
 - Traverse the vector BACKWARD from $p[t]$ and report each point $// \mathcal{O}(r)$
until escaping from I at point $p[s]$
 - return $r = t - s$ $//$ output size



Output-Sensitivity

- ❖ An **enumerating** query can be answered in $\mathcal{O}(r + \log n)$ time
- ❖ $p[s]$ can also be found by binary search in $\mathcal{O}(\log n)$ time
- ❖ Hence for COUNTING query, $\mathcal{O}(\log n)$ time is enough //independent to r
- ❖ Can this simple strategy be extended to PLANAR range query?

TTBOMK, unfortunately, no!

