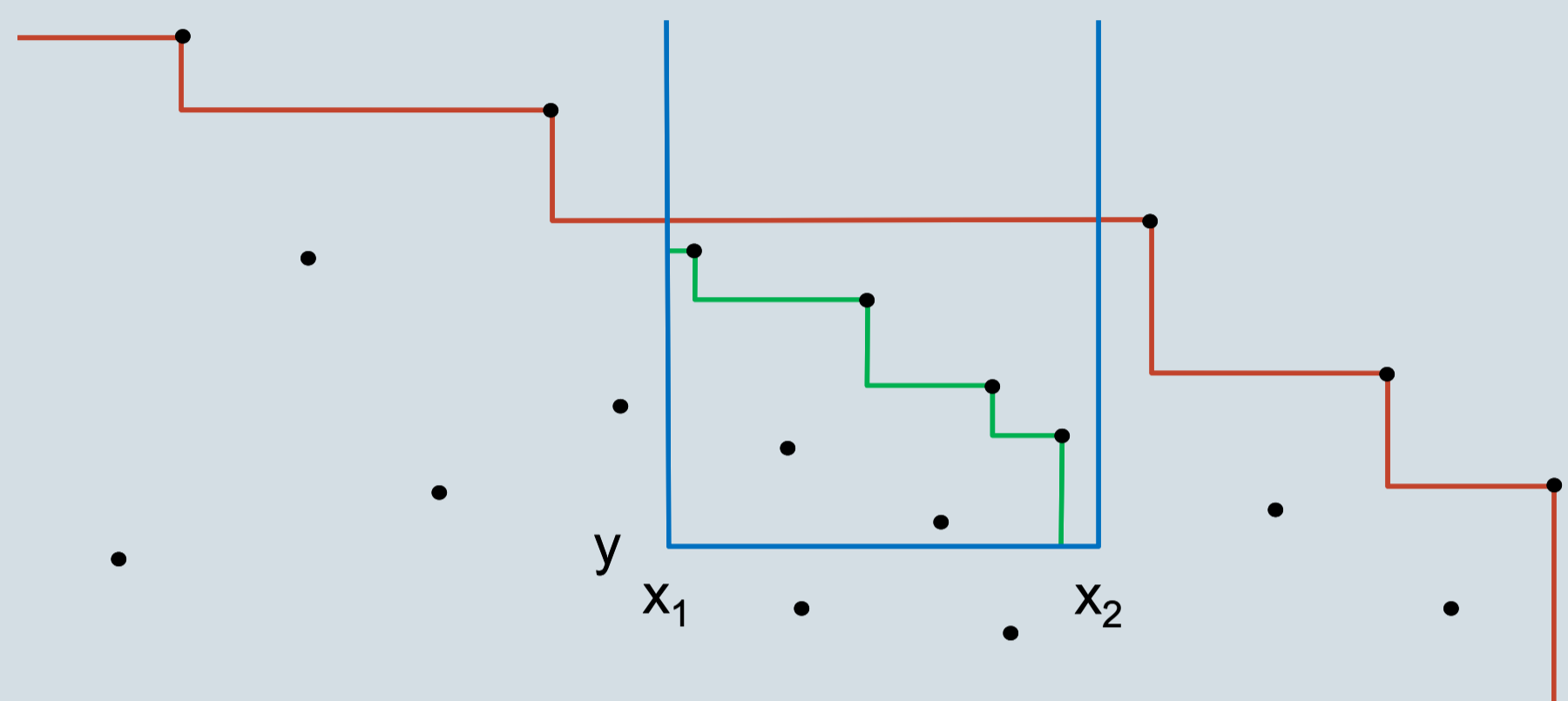


I/O-Efficient Dynamic Planar Range Skyline Queries

Problem Definition

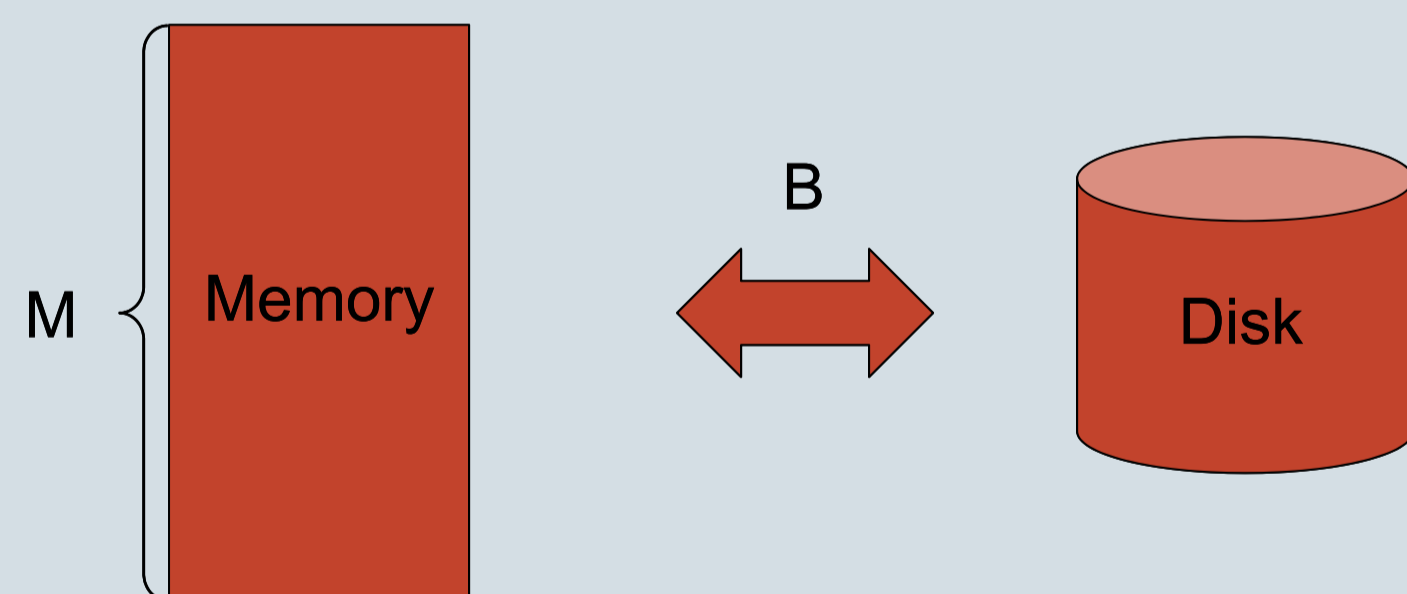
We study the problem of maintaining a planar point set $P \subseteq \mathbb{R}^2$ under the modifications of insertion and deletion. Given two points $p, q \in \mathbb{R}^2$ we say that p *dominates* q iff $x(p) \geq x(q) \wedge y(p) \geq y(q)$, i.e. all coordinates of p are larger than those of q . The maximal points of $S \subseteq P$ are all the points of S that are not dominated by any point in S .



We want to support 3-sided orthogonal skyline reporting queries for the point set P , i.e. reporting all maximal points in $S \cap P$ where $S = [x_1; x_2] \times [y; \infty]$.

The I/O Model

In the I/O model we have a memory capable of holding M elements and a disk of infinite size. When we read or write to the disk we can read or write B elements at a time, and the cost is 1 I/O. The cost is the same if we read or write less than B elements at a time, hence we want to batch reads and writes together.



Previous / Our Results

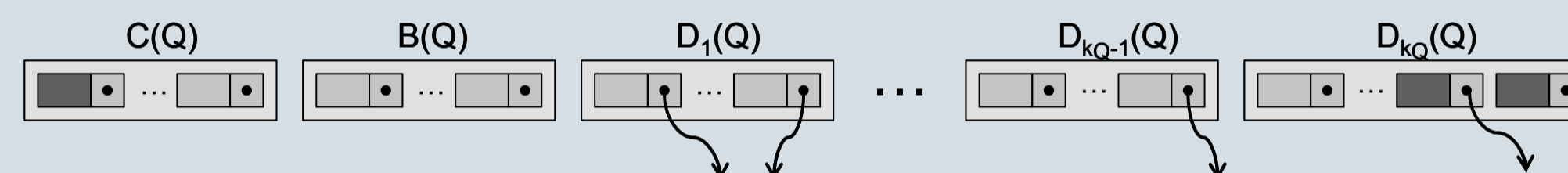
Our result is the first dynamic orthogonal skyline reporting data structure in the I/O model, all previous results are either not fully dynamic or are in a different model.

Reference	Model	Insert / Delete	Query
[BT 2011]	Pointer Machine	$O(\log n)$	$O(\log n + t)$
[BT 2011]	RAM	$O(\log n / \log \log n)$	$O(\log n / \log \log n + t)$
[KTT 2013]	I/O	$O(\log_{2B^\epsilon} n)$	$O(\log_{2B^\epsilon} n + t/B^{1-\epsilon})$

I/O-CPQA

The I/O efficient Catalanable Priority Queue with Attrition (I/O-CPQA) supports the following operations in $O(1)$ I/Os worst-case and $O(1/B)$ I/Os amortized:

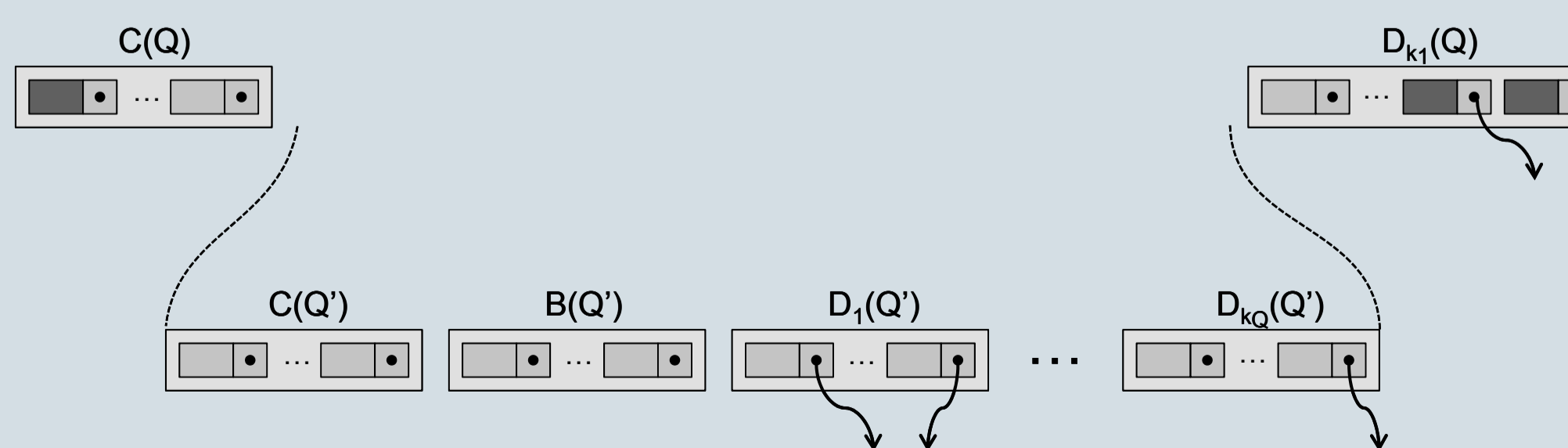
- **InsertAndAttrite(Q,e)** – appends e to the end of Q and deletes all attrited elements in Q that are smaller than e , i.e. returns $Q' = \{e' \in Q \mid e' < e\} \cup \{e\}$.
- **DeleteMin(Q)** – deletes the minimum element $e = \min(Q)$ from Q and returns e and $Q \setminus \{e\}$.
- **ConcatenateAndAttrite(Q₁,Q₂)** – Deletes all attrited elements in Q_1 that are smaller than $e = \min(Q_2)$ and prepends the non-attrited elements onto Q_2 , i.e. returns $Q' = \{e' \in Q_1 \mid e' < e\} \cup Q_2$.



A *record* consists of a buffer of $[b; 4b]$ elements and a pointer to another I/O-CPQA Q where $\max(b) < \min(Q)$. An I/O-CPQA Q consists of $2+k_Q$ functional deques $C, B, D_1, \dots, D_{k_Q}$ of records. All records in the deque fulfill $\max(p) < \min(q)$ where record p precedes record q in the same deque. Also $\max(C) < \min(B) < \min(D_1)$ and $\min(D_1)$ is the smallest element in all of the dirty deques D_i . The essential size invariant to guarantee the bounds is:

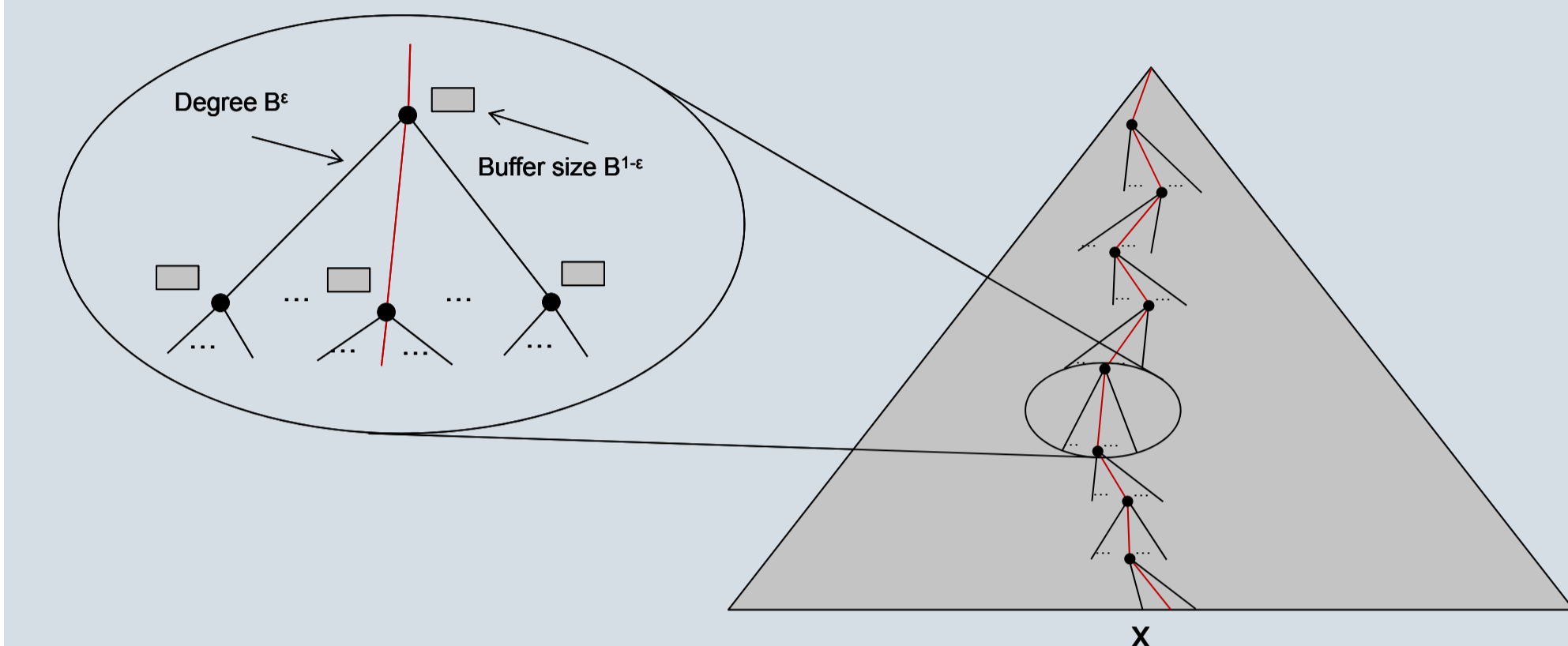
$$|C(Q)| \geq \sum_{i=1}^{k_Q} |D_i(Q)| + k_Q - 1 \quad (*)$$

When we delete the minimum element from the I/O-CPQA we delete from C , if this violates (*) then we take records out of B and put them into C , else if $k_Q > 1$ we merge D_{k_Q-1} and D_{k_Q} else we put the first record v of D_1 into C and we put the I/O-CPQA Q' that v points to into Q , see the figure below.

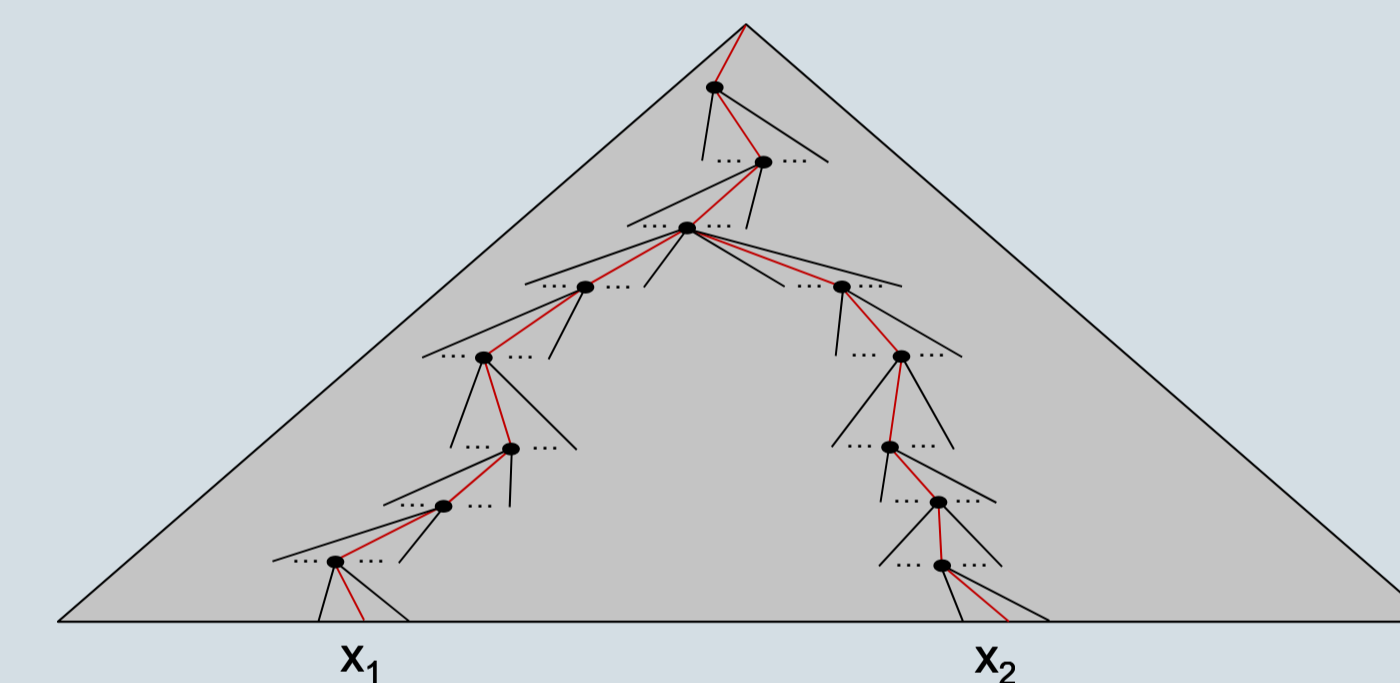


2D I/O Maxima

We build a $(2B^\epsilon; 4B^\epsilon)$ -tree with each leaf being an I/O-CPQA of $B^{1-\epsilon}$ elements and each internal node stores the concatenation of its children's I/O-CPQAs. This takes $O(n)$ space since the I/O-CPQAs of the internal nodes will only use $O(1)$ blocks of space as they are functional.



When we receive an update we find the position in the tree and discards all I/O-CPQAs on the same path and reconstruct the path bottom up.



When we do a 3-sided range query for $[x_1; x_2] \times [y; \infty]$ we concatenate the $O(B^\epsilon \log_{2B^\epsilon} n)$ I/O-CPQAs in $O(\log_{2B^\epsilon} n)$ I/Os and then call **DeleteMin** on the resulting I/O-CPQA until the minimum element m returned is $m < y$, this then takes $O(t/B^{1-\epsilon})$ I/Os.

Future Work

It is still an open problem whatever it is possible to obtain bounds of $O(\log_B n)$ for updates and $O(\log_B n + t/B)$ for queries, or it is possible to show a lower bound. The ϵ in our bounds comes because we need to load B^ϵ I/O-CPQAs and concatenate them in $O(1)$ I/Os, which we are only able to do if the buffer size of each record is $B^{1-\epsilon}$.

References

- [KTT 2013] Kejlberg-Rasmussen, Tsakalidis, Tsihlias. *I/O-Efficient Dynamic Planar Range Skyline Queries*. Submitted.
- [BT 2011] Brodal, Tsakalidis. *Dynamic Planar Range Maxima Queries*. ICALP 2011.