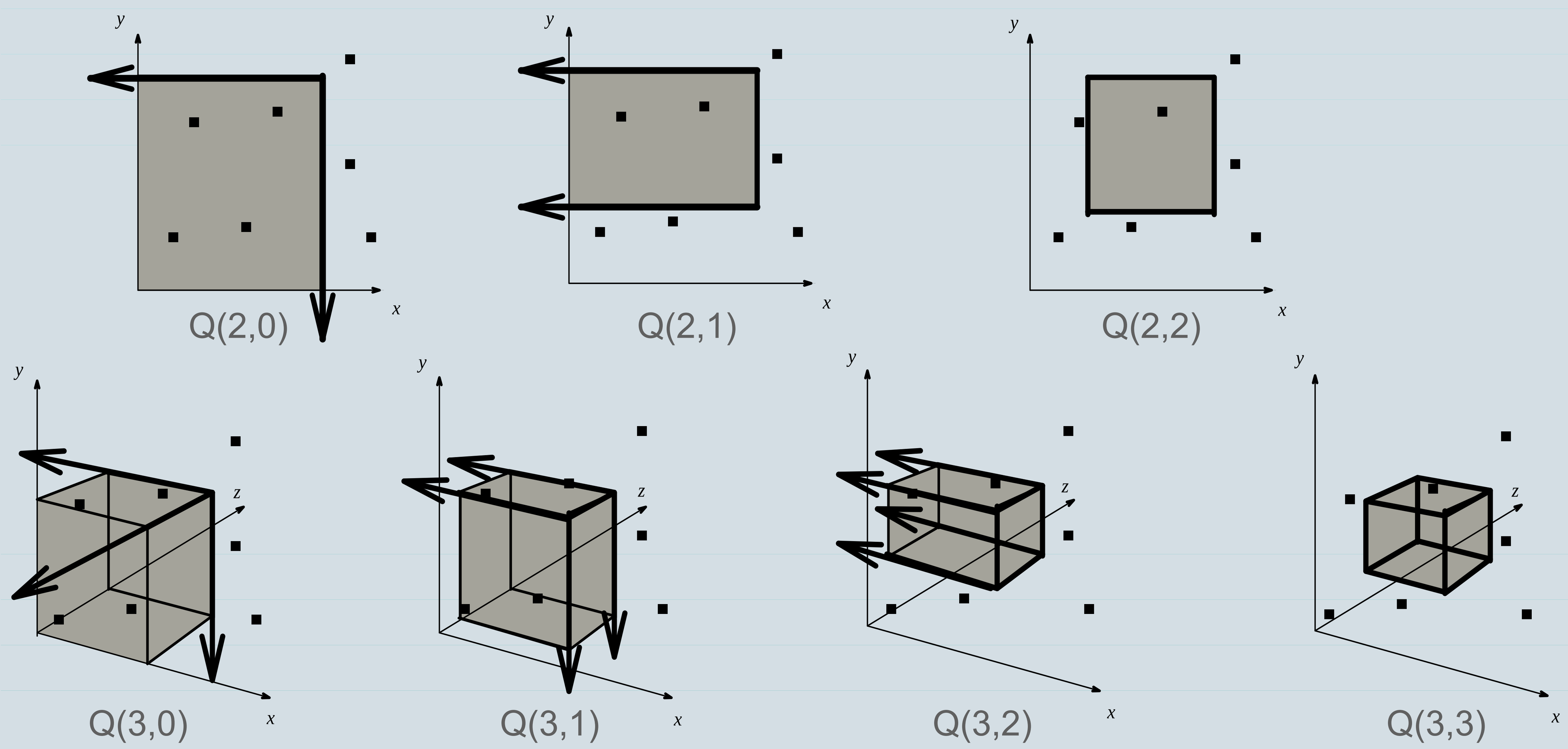
MADALGO
CENTER FOR MASSIVE DATA ALGORITHMICS

Kasper Dalgaard Larsen
Aarhus University

Three-Dimensional Range Search Indexing

Problem

Preprocess a set of N 3-dimensional points into an I/O-efficient data structure, such that all points inside an axis aligned query hyper rectangle can be reported efficiently.



Q(d,k) denotes the d dimensional problem with k finitely bounded dimensions.

Results

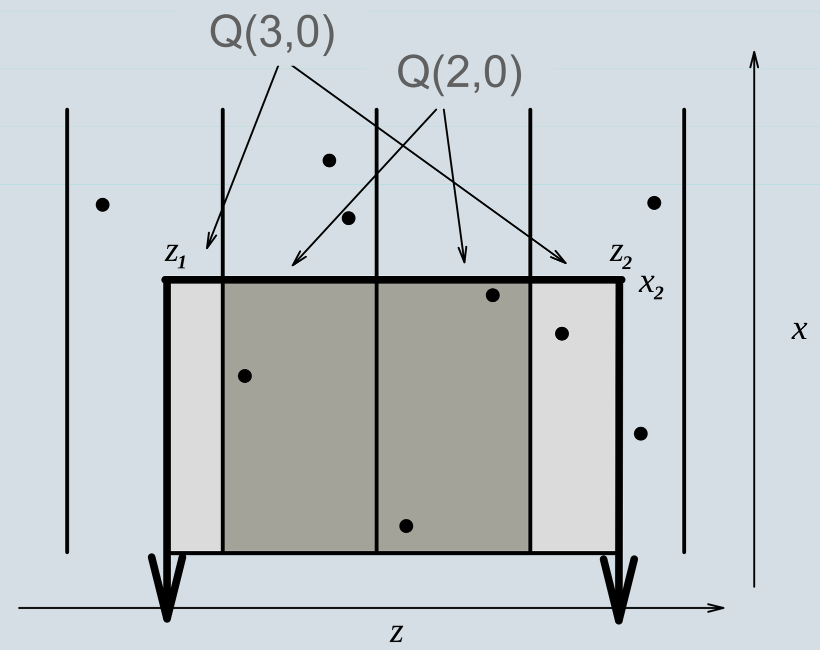
Problem	Previous Work	Our Contribution[2]
Q(2,0)	O(N) [3]	-
Q(2,1)	O(N) [3]	-
Q(2,2)	$O(N(\log_2 N / \log_2 \log_B N))$ [3]	-
Q(3,0)	O(N) [1]	-
Q(3,1)	$O(N(\log_2 N))$ [1]	$O(N(\log_2 N / \log_2 \log_B N))$
Q(3,2)	$O(N(\log_2 N)^2)$ [1]	$O(N(\log_2 N / \log_2 \log_B N)^2)$
Q(3,3)	$O(N(\log_2 N)^3)$ [1]	$O(N(\log_2 N / \log_2 \log_B N)^3)$
Q(d,d) lower bound	$\Omega(N(\log_2 B / \log_2 \log_B N)^{d-1})$ [4]	$\Omega(N(\log_2 N / \log_2 \log_B N)^{d-1})$

Space usage of previous and new structures. All have optimal $O(\log_B N+T/B)$ query cost, where B is the I/O block size and T is the output size.

Data Structures

Basic Structure

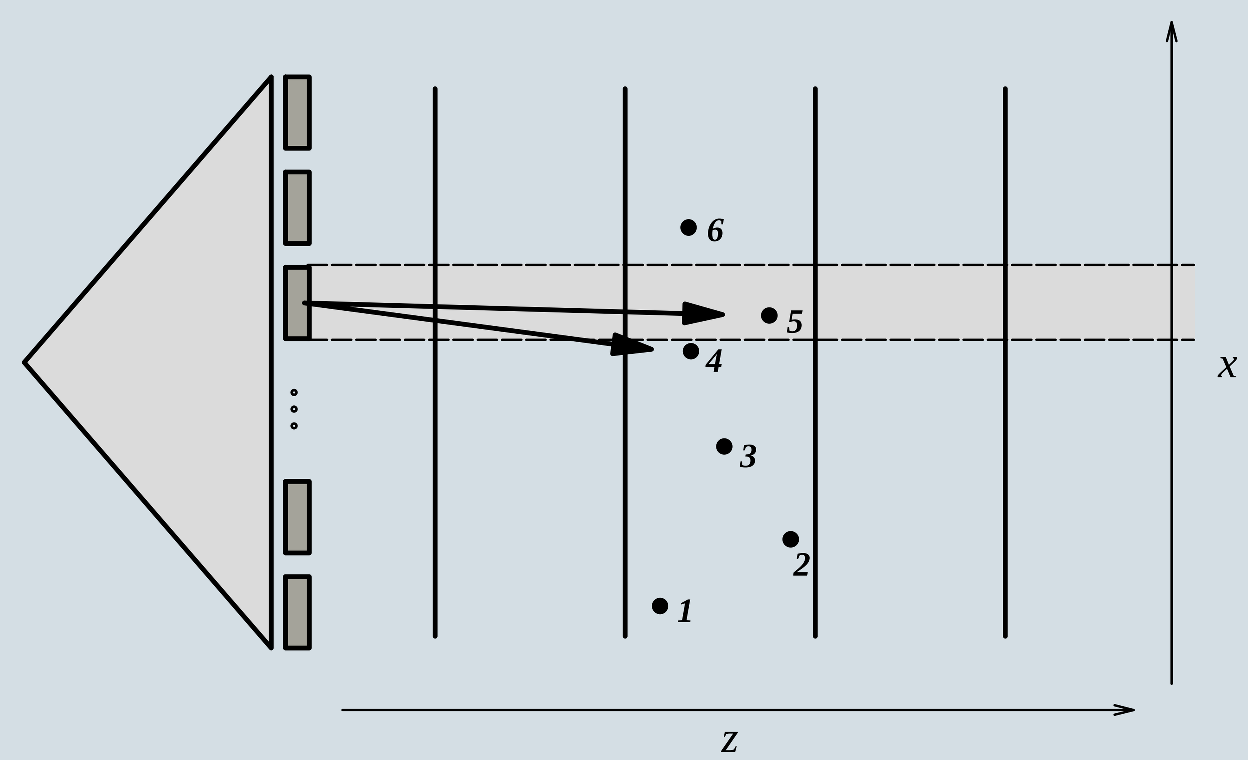
Our Q(3,k) structure (k>0) is a search tree over one of the k finitely bounded dimensions. The tree has fanout $(\log_B N)^{(1-\varepsilon)/2}$.



Queries

To answer a query *q*, search down the tree until *q* intersects several children of a node, *q* is then decomposed into two Q(3,k-1) queries and $O((\log_B N)^{(1-\varepsilon)/2})$ Q(2,k-1) queries. This is shown for Q(3,1) above.

The Q(3,k-1) queries are answered in a Q(3,k-1) structure stored in the children, but the Q(2,k-1) queries cannot be answered this way, since there are $O((\log_B N)^{(1-\varepsilon)/2})$ of them.



For Q(3,2) and Q(3,3), we store a Q(2,k-1) structure for every range of children $c_i \ldots c_j$. The trick is to answer all Q(2,k-1) queries at the same time, and we do this by querying the correct such structure.

For Q(3,1) we develop a structure answering Q(2,0) $[-\infty,x_1]\mathrm{x}[-\infty,y_1]$ queries in O(1+T/B) I/Os if we know the rank of x_1 amongst the points. We then build a rank tree as shown above, which obtains the rank of x_1 in each child at the same time.

Lower Bound

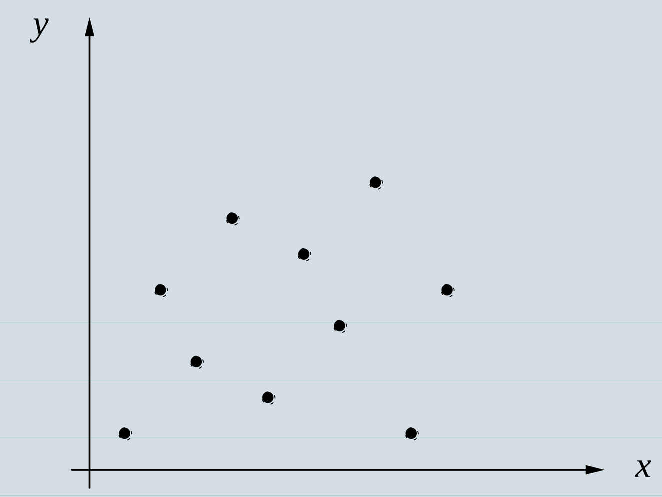
Indexability Model

Construct a hard d-dimensional point set and query set.

Point set

$$P = \{(p_{a(1)}(i), p_{a(2)}(i), \ldots, p_{a(d-1)}(i), i) \mid i=0,\ldots,N-1\}$$

$p_{a(j)}(i)$ is *i* written in base a(j) truncated to $\log_{a(j)} N^{(1/d)}$ digits, and finally with those digits reversed. The a(j)'s are d-1 relatively prime integers all greater than $\log_B N$.



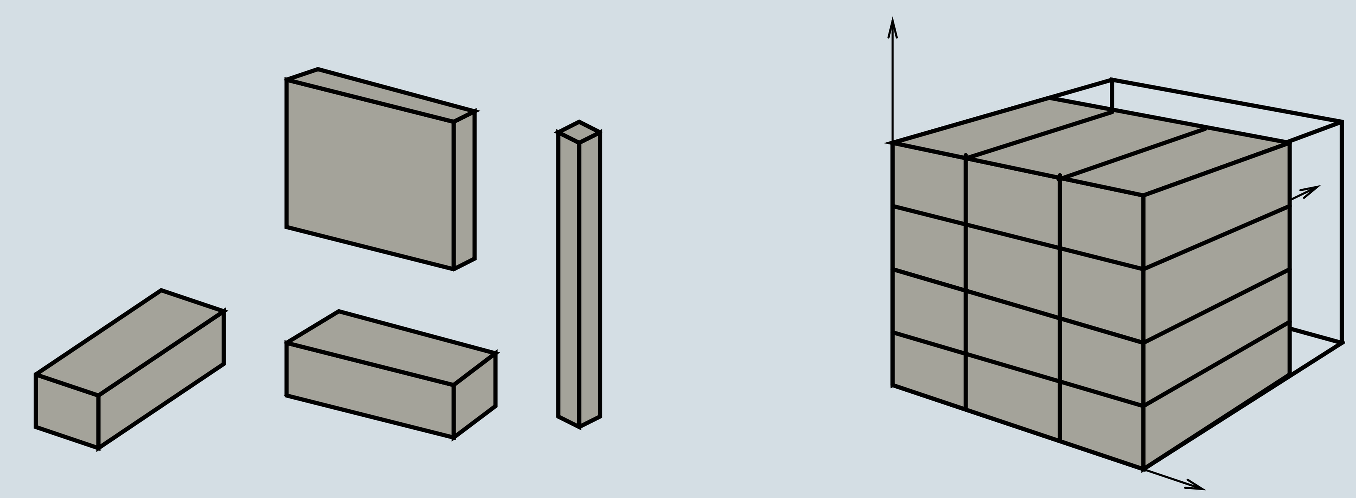
Hard point set for 2D

Query set

Queries are defined from a set of candidate query dimensions

$$(\log_{a(1)} N)^{k(1)} \times \cdots \times (\log_{a(d-1)} N)^{k(d-1)} \times B \log_B N\, f(k(1),\ldots,k(d-1))$$

where $k(j) \in \{0,\ldots, \log_{a(j)} N^{(1/d)}\}$. For each candidate, we tile the point set with query rectangles of that dimension.



References

[1] P. Afshani. On dominance reporting in 3D. In *Proc. European Symposium on Algorithms*, pages 41-51, 2008.
[2] L. Arge and K. D. Karsen. Towards Optimal Three-Dimensional Range Search Indexing. *Submitted.*
[3] L. Arge, V. Samoladas and J. S. Vitter. On two-dimensional indexability and optimal range searching. In *Proc. ACM Symposium on Principles of Database Systems*, pages 346-357, 1999.
[4] J. Hellerstein, E. Koutsoupias, D. Miranker, C. Papadimitriou and V. Samoladas. On a model of indexability and its bounds for range queries. *Journal of ACM*, 49(1), 2002.

MADALGO – Center for Massive Data Algorithmics, a Center of the Danish National Research Foundation