3D range minimum queries

Nicola Prezza

References and Reading

[1] Sections 7.2, 7.4.1 of: Navarro, Gonzalo. Compact data structures: A practical approach. Cambridge University Press, 2016.

Exercises

1 3D minimum range queries As seen in Section 7.4.1, $2n+o(n)$ bits are sufficient to determine, in $O(\log\log n)$ time, the index of a minimum element in any range of a given input array $A[1,n]$ of integers (more advanced solutions achieve even constant time within the same space). Note that this can be viewed as a geometric problem on a two-dimensional grid: given a set of points $\langle 1, y_1\rangle, \dots, \langle n, y_n\rangle$, return the x-coordinate of a point with minimum y-coordinate in a given x-range $[X, X+l]$.

We want now to generalize this problem to three dimensions. The input of our problem consists of a list of n 3D points $\langle 1, y_1, z_1\rangle, \dots, \langle n, y_n, z_n\rangle$ on the 3D cube $[1,n]\times[1,n]\times[1,n]$, where $1 \le y_i, z_i \le n$ for all $i = 1,\dots,n$ (note that x-coordinates are precisely $1,2,\dots,n$). We want to build a data structure as space-efficient as possible[1] supporting efficiently the following query: given a four-sided range on x-y coordinates, return the x coordinate of a point with minimum z-coordinate in the range (if there is more than one such point, choose arbitrarily). More formally: given a x-y range $[X, X+l_x]\times[Y, Y+l_y]$, return j, with $X \le j \le X+l_x$ and $Y \le y_j \le Y+l_y$, such that $z_j \le z_k$ for all k satisfying $X \le k \le X+l_x$ and $Y \le y_k \le Y+l_y$.

Example

				4	
6					
			5		
	2				3
		1			

The example depicts a cube $[1,6]\times[1,6]\times[1,6]$ populated with the 6 points $\langle 1,5,6\rangle$, $\langle 2,3,2\rangle$, $\langle 3,1,1\rangle$, $\langle 4,4,5\rangle$, $\langle 5,6,4\rangle$, $\langle 6,3,3\rangle$ (note: x-y coordinates are represented as usual on a square, while z coordinates are encoded in the numbers stored in the cells). These are the results of some example queries:

- Input: $[2,4]\times[2,5]$. Output: 2
- Input: $[2,4]\times[1,5]$. Output: 3
- Input: $[3,6]\times[2,6]$. Output: 6

[1] note that, since x-coordinates are $1,\dots,n$, the information-theoretic minimum number of bits needed to store such a cube is $2n\log n$. Try to get as close as possible to this quantity.