6.897 Advanced Data Structures (Spring'05)

Prof. Erik Demaine TA: Mihai Pătraşcu

$\boxed{\text{Problem 9}}$ *Due: Friday, Apr. 15*

This problem shows the remarkable connections between different problems that we studied in this course, and represents a coronation a large body of research.

Reread the last problem if you're not familiar with the following notions: existential range queries, rank space, three-sided queries, dominance queries. In the last problem, we showed that static existential range queries in rank space can be solved very efficiently: constant query time and $O(n \lg n)$ space. We also said that if the problem is not in rank space, we can run 4 predecessor queries, and reduce it to rank space. However, predecessor queries take superconstant time, so we need to ask whether this is really necessary (maybe one can solve the problem faster without explicitly reducing it to rank space). In this problem, we show that any lower bound for the colored predecessor problem also applies to two-dimension range queries. Thus, reducing to rank space via predecessor queries is optimal. Also note that predecessor queries in a universe of size $O(n)$ take constant time (table lookup), so we are not contradicting our result for rank space.

Consider the following segment stabbing problem. We are given a set of closed segments on the line, $S = \{[a_i, b_i] \mid i = 1..n\}$. A query is of the form: given x, does x stab any segment? That is, $\exists\ [a,b] \in S : x \in [a,b]$?

1. Argue that the static colored predecessor problem reduces to static segment stabbing (if you can solve the latter, you can also solve the former in the same time and space bounds).

Remember that the colored predecessor problem considers a set of points colored red or blue, and asks for the color of the predecessor of some x.

2. Argue that segment stabbing reduces to existential dominance queries in two dimensions. Assume coordinates are integers in $\{0, \dots, u\}$.

This shows that the lower bounds for predecessor also apply to existential range queries. Note that we are reducing to the least general type of range queries (dominance), so the lower bound is as strong as possible.

Up until now, we have only considered static problems – but how about the dynamic case? One can solve dynamic RMQ (thus, also 3-sided queries) in $O(\frac{\lg n}{\lg \lg n})$ per operation. For no good reason, dynamic RMQ is usually called "priority range searching". Unfortunately, the transformation from 3-sided to general queries from problem 8 only works in the static case. However, it was recently shown, through much more complicated techniques, how to obtain the same $O(\frac{\lg n}{\lg \lg n})$ bound for general queries. Observe that these data structures are insensitive to the rank space vs general coordinates issue, because we can solve dynamic predecessor queries in $O(\sqrt{\frac{\lg n}{\lg \lg n}})$.

3. Argue that the dynamic marked ancestor problem reduces to dynamic segment stabbing.

Remember that the dynamic marked ancestor problem considers a *static* arbitrary tree on n nodes (which can be preprocessed). An update can mark or unmark any nodes; a query is: given a leaf, is there any marked node on the root-to-leaf path?

This gives a rather amazing relation between a problem on trees and a geometric problem. In class, we described an $\Omega(\frac{\lg n}{\lg \lg n})$ lower bound for the marked ancestor problem. Thus, we have shown that the above-mentioned data structures for dynamic range reporting are optimal.