One-Dimensional Range Counting Queries using Range Trees [1]

Range trees can also be used to answer other types of queries. We can apply the range tree approach to one-dimensional and multi-dimensional problems. In this note I will show how one-dimensional range counting queries can be answered in $O(\log n)$ time. The problem is to store a set of two-dimensional points in a data structure, so that for any range $[x_1, x_2]$ we can *count* the *number* of points whose x-coordinates are in $[x_1, x_2]$. We use the same example as in the note on range trees by Shahin Kamali.

$p_0 : (8, 47)$ $p_1 : (7, 21)$ $p_2 : (9, 15)$ $p_3 : (11, 25)$ $p_4 : (13, 40)$
$p_5 : (14, 34)$ $p_6 : (15, 14)$ $p_7 : (16, 31)$ $p_8 : (18, 17)$ $p_9 : (19, 36)$
$p_{10} : (10, 8)$ $p_{11} : (5, 12)$ $p_{12} : (1, 28)$ $p_{13} :: (4, 5)$ $p_{14} : (3, 45)$
$p_{15} : (20, 3)$ $p_{16} : (22, 16)$ $p_{17} : (24, 46)$ $p_{18} : (28, 1)$ $p_{19} : (30, 19)$
$p_{20} : (35, 50)$ $p_{21} : (32, 18)$ $p_{22} : (40, 6)$ $p_{23} : (42, 20)$ $p_{24} : (48, 2)$
$p_{25} : (50, 41)$ $p_{26} : (61, 55)$ $p_{27} : (63, 24)$ $p_{28} : (74, 70)$ $p_{29} : (90, 27)$
$p_{30} : (80, 31)$

Figure 1 shows the main balanced BST based on x-coordinates of points. We do not need any secondary trees in this case. Instead we keep in every node v the total numbern_v of points in its subtree (i.e., the total number of points in v and all its descendants). This information is sufficient to answer one-dimensional range counting queries. Given a query $[x_1, x_2]$, we search for x_1 in T and we search for x_2 in T. Let P_1 denote the search path for P_1 and let P_2 denote the search path for P_2. We examine all boundary nodes (i.e., nodes on P_1 or P_2) and count the number n_1 of points in $[x_1, x_2]$ that are stored in boundary nodes v on $P_1 \cup P_2$. We also examine all top inside nodes and count $n_2 = \sum n_v$ where the sum is taken over all top inner nodes. Then the total number of points in $[x_1, x_2]$ is equal to $n_1 + n_2$. For instance, suppose that we want to count the number of points in $[4, 59]$. We examine all boundary nodes and compute n_1, the number of points in boundary nodes that are in $[x_1, x_2]$. In our example, $n_1 = 6$. Top inside nodes are nodes that contain p_0, p_6, p_{19}, and p_{24}. The total number of points in or below these four nodes is $n_2 = 18$. Hence there are $n_1 + n_2 = 24$ points with x-coordinates in $[4, 59]$.

[1]Prepared for CS 240 - University of Waterloo - Spring 2015 (S. K., Y.N., O. Z.)

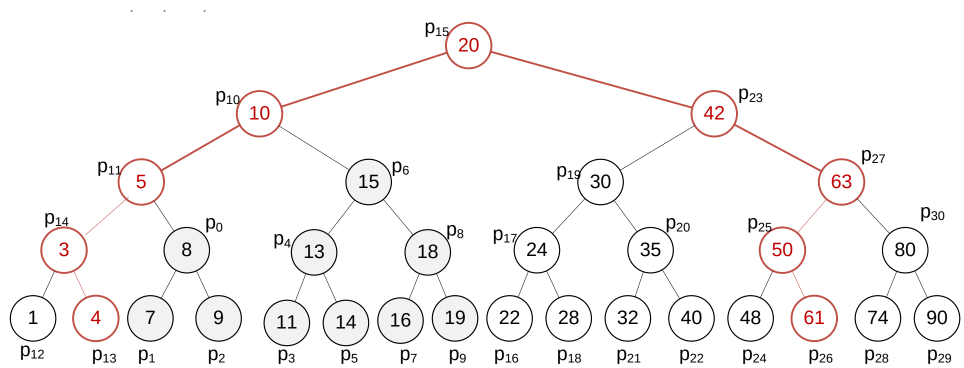

Figure 1: The range tree of the listed points. The red nodes indicate the boundary points for range $4 \leq x \leq 59$.