Segment tree
Theory and applications

Gleb Evstropov
glebshp@yandex.ru

November 12, 2015

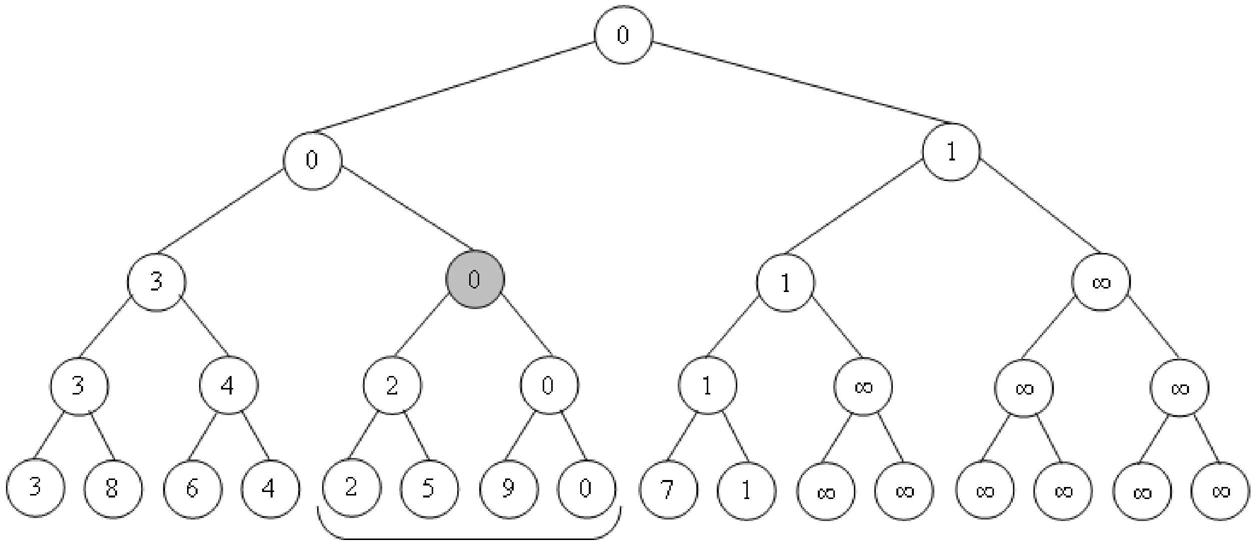
Special thanks to Maxim Akhmedov for providing illustrations.

- *RMQ* stands for Range Minimum (Maximum) Query problem;
- *RSQ* stands for Range Sum Query problem;
- Problem is called *dynamic* if there are `Change` queries;
- Problem is called *static* if there are no `Change` queries;
- Binary operation $\oplus$ is called *associative* if it satisfies the associative law:

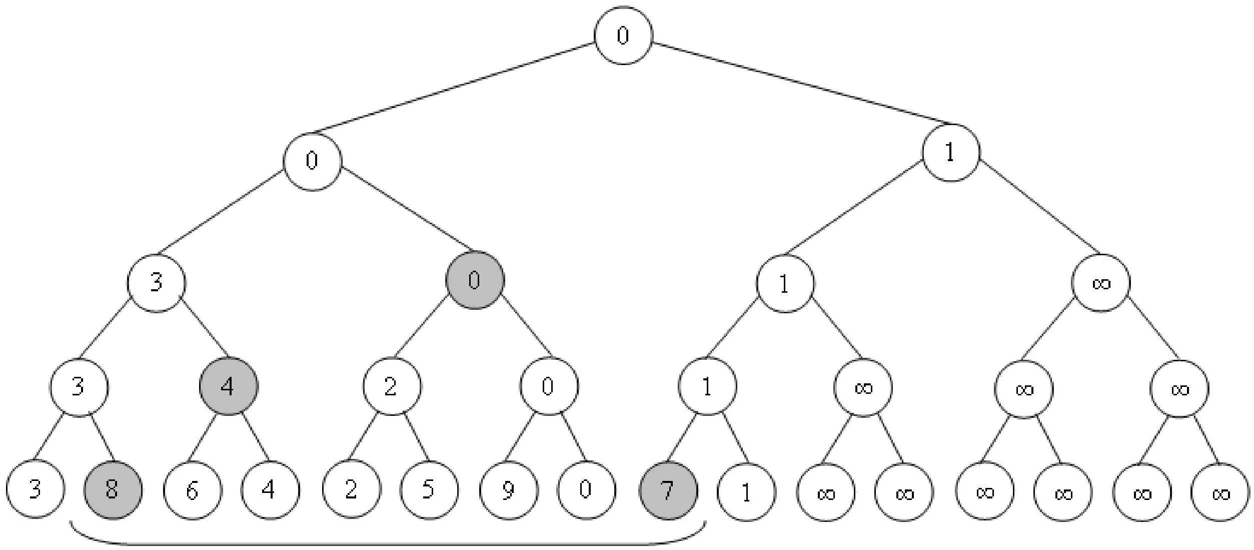
$$(a \oplus b) \oplus c = a \oplus (b \oplus c)$$

- Binary operation $\oplus$ is *idempotent* if $a \oplus a = a$;
- *Identity element* for some pair $(S, \oplus)$ is such an element $e \in S$ that for every $a \in S$ condition $a \oplus e = e \oplus a = a$ holds;
- A *semigroup* is an algebraic structure consisting of a set S together with some associative binary operation $\oplus$;
- *Monoid* is a semigroup with an identity element;
- For every array a_i, where every element belongs to some monoid $(S, \oplus)$ we can build a segment tree to answer the following queries:
  - `Get(l, r)` — returns $a_l \oplus a_{l+1} \oplus \ldots a_r$
  - `Change(p, x)` — set $a_p = x$
- For simplicity we will supplement the array a_i with identity elements e in order to make it's length equal to some power of two, i.e. $n = 2^k$ for some integer k;

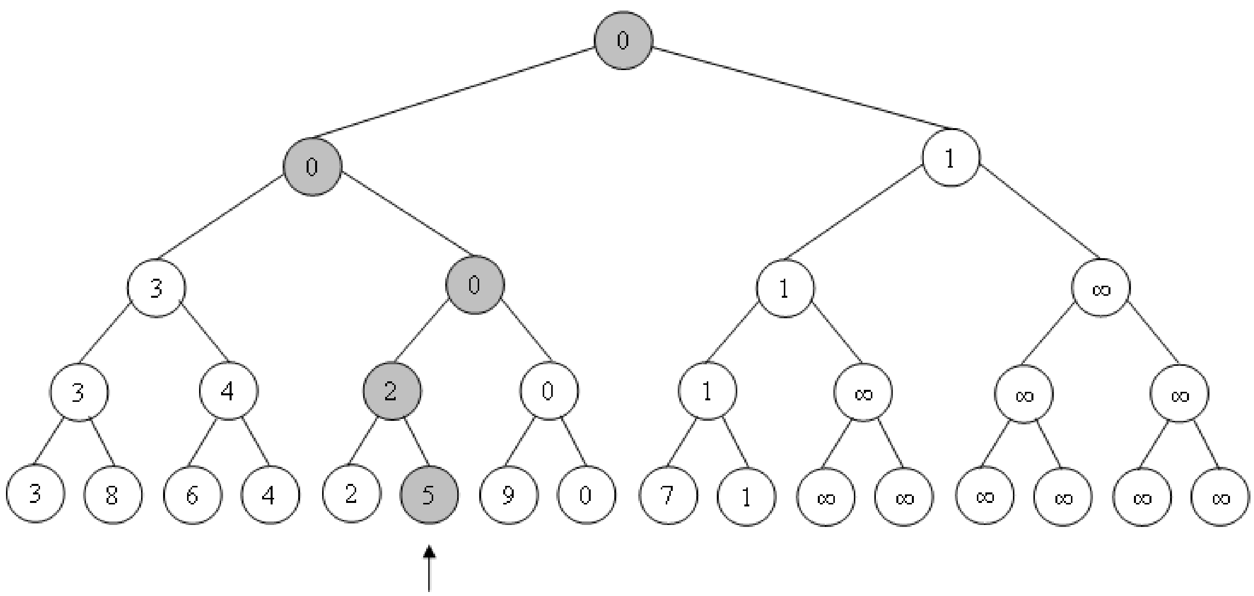
- Segment tree stores accumulated values for all segments of length 1, $n/2$ non-overlapping segments of length 2, $n/4$ non-overlapping segments of length 4 and so on. Such segments are called *fundamental*;
- Each fundamental segment may be treated as a node of the tree:



- The good way to store a tree is an array with indices starting from 1. Then the left child of a stored at position i has index $2 \cdot i$ and the right child has index $2 \cdot i + 1$. The total number of nodes is $2^{k+1} - 1$ with 1 being a root and leaves stored at positions from 2^k to $2^{k+1} - 1$;
- Main property of the set of fundamental segments: every segment $(l,\ r)$ can be represented as a union of no more than $2 \cdot \log(n)$ non-overlapping fundamental segments;
- Rule to choose fundamental segments: take a segment $(i,\ j)$ in decomposition of a segment $(l,\ r)$ if and only if $(i,\ j)$ is a subsegment of $(l,\ r)$ and the parent of $(i,\ j)$ in the tree is not a subsegment of $(l,\ r)$;



- To apply `Change` operation one needs to traverse a path from leaf to root:



- Binary operation $\cdot$ is right-distributive over $\oplus$ if $(a \cdot c) \oplus (b \cdot c) = (a \oplus b) \cdot c$;

- One-dimensional segment tree with group updates can be used if both elements a_i form monoid for binary operation $\oplus$, all updates form monoid for binary operation $\cdot$ and $\cdot$ is right-distributive over $\oplus$.