1. Given a configuration c.

Ex. c = [50, 50, 50, 50]

c[i] is ratio of $\frac{n_r[i]}{n_g[i]}$

where $n_r[i]$:= # of reads that match to ith red exon, $i \in \{1, 2, 3, 4, 5\}$

and similarly green ⇒ $n_g[i]$.

The prob of ~~given~~ observing data given config is then product of prob of observing counts for each exon independently

$$p(\text{Exon}[i] \mid \text{Config}, \text{Data}) = {}^{n^*_r[i] + n^*_g[i]}C_{n^*_r[i]} \, (p_i)^{n^*_r[i]} (1-p_i)^{n^*_g[i]}$$

where $p_i = \frac{n\_r[i]}{n\_r[i] + n\_g[i]} = \frac{c[i]}{c[i] + 100}$

where $n^*r[i]$ = # of reads that actually match to red
$n^*\_g[i]$ = # of " " " " " green exon i

So, from prob generated from program,

C_3 = [33%, 33%, 100%, 100%] seems to be the best possible explanations.

Given a Binary Array of size n. Answering Select (r) queries.

Data Structures : Sel (x) data structure. Given rank x, Sel(x) returns the index of xth one in the Array. Ex.

0	1	1	1	0	1	1	1	1

Ex. Sel (2) = 2 ; Sel (3) = 4 etc.

Skip Data Structure : Skip (x) : Given index x, stores the index of next 1 in Array.

Example

1	2	4	4	5	∞

Skip

dummy.

Algo : Select(r)

- $r\_low = \lfloor \frac{r}{\Delta} \rfloor$
- $r\_up = \lceil \frac{r}{\Delta} \rceil$
- cur_rank = sel(r_low) , i = r_low {inclusive}
- while i <= r_up
  - if cur - rank == r
    return i
  - i = Skip(i) // move to next 1.
  - cur - rank += 1.

Time Complexity : $O(\Delta)$ when Sel is milestoned at Δ intervals, since the loop can run at max Δ times. This is so because there are no more than Δ 1s between Sel(r-low) and Sel(r-up) indices and using the skip DS we skip to next 1 in $O(1)$ time. (milestoned at Δ intervals).

Space : $\left(\frac{n}{\Delta}\right)$ to store Sel

$O(k)$ to store skip where k is the number of 1s in Binary array. Note that we do not need to store Skip values at 0s. If $k > \frac{n}{2}$, we can develop a complementary algo using 0s. Hence $k \le \frac{n}{2}$ $\therefore$ $\frac{n}{2} + \frac{n}{\Delta} = c\frac{n}{\Delta}$ $\therefore$ $c \le \left(\frac{\Delta}{2} + 1\right)$.