1. The operations are of the form $ax + b$, we can maintain them using multiplicative inverse. $O(\log P)$ for getIndex, and $O(1)$ for other operation, or $O(\log P)$ for append, and $O(1)$ for other operation.
2. Segment tree. $O(\log n)$ for getIndex, and $O(1)$ for other operation.
3. Union find, and only use path compression. $O(n \log n)$.
4. We also want to use union by rank in the union find data structure, but if we union in the reverse direction, we need to compute multiplicative inverse, which is costly. As a solution, let b be a parameter to be set later, we divide the input stream into blocks with size b. After reading each block, we can compute the multiplicative inverse in $O(\log P)$ time, and we can for each element in the block compute the merge of operations from it to the end of the block, in $O(b)$ time total. Represent each block with a single node in the union find structure. As long as $b \geq \log \log n$, the union-find data structure has running time $O(n\alpha(n, \frac{n}{b})) = O(n\alpha(n, \frac{n}{\log \log n})) = O(n)$. For queries in the (partial) last block, we can recursively use an algorithm with running time $O(\log b)$. (can we bootstrap?) Set $b = \frac{\log P}{\log \log P}$ (which always $\geq \log \log n$, because $P \geq n$, otherwise we can compute multiplicative inverse in amortized $O(1)$ time), the total running time is $O(n \log \log P)$.
When we recurse, group into miniblocks with size $O(\log b)$, and use algorithm 3 (only path compression), with running time $O(\log_{1+\frac{b}{b/\log b}} \frac{b}{\log b}) = O(\frac{\log b}{\log \log b})$ per operation (by Jensen's inequality, the worst case is each block has $O(b)$ queries on average). Set $b = \frac{\log P \log \log \log P}{\log \log P}$, the total running time is $O(n \frac{\log \log P}{\log \log \log P})$.

Remark. can we set the block size to be $b' = \sqrt{n'}$ and bootstrap (so that the complexity of path compression is $O(\log_{1+b'} n') = O(1)$), and get $O(n \log \log b) = O(n \log \log \log P)$ time? Improvable to $O(n\alpha(n))$?
Is the running time of union-find data structure improvable in this case?

References