Optional Problems 1: Integer Data Structures

These problems are purely optional but might help you get a better understanding of the data structures we've covered in Week 8. Feel free to work through them if you'd like!

Problem One: Reducing vEB Space Usage

(This problem adapted from a problem used in MIT's Advanced Data Structures course.)

van Emde Boas trees require $\Theta(U)$ machine words of storage regardless of space usage. This makes them prohibitively large in many applications.

It's possible to shrink this space usage down to $\Theta(U)$ *bits* of storage by adding in some indirection along the lines used by a *y*-fast trie. (This is historically backwards; the *y*-fast trie was motivated by an indirection technique developed for vEB trees.)

Show how to reduce the space usage of a vEB tree to $\Theta(U)$ bits while supporting all operations in time $O(\log \log U)$. As a hint, make an auxiliary bitvector of size $\Theta(U)$ and split the universe into blocks of size $\log w$, where w is the machine word size.

Problem Two: *x*-Fast and *y*-Fast Tries

i. In lecture, we glossed over the details of how to implement deletion in an *x*-fast trie. Describe how to implement this operation in expected, amortized time $O(\log U)$.

ii. Describe how to update a *y*-fast trie so that ***min*** and ***max*** run in time $O(1)$.

Problem Three: Improved Graph Algorithms

This problem considers the application of our integer data structures to a variety of graph algorithms.

i. Design an $O(U + m \log \log U)$-time, deterministic algorithm for finding a minimum spanning tree of an undirected graph in which all edge weights are in $[U]$. For reference, there is a known algorithm due to Fredman and Willard that computes a minimum spanning tree on graphs with integer edge weights in time $O(m + n)$ assuming that each integer fits into a machine word.

ii. Design an $O(U + m \log \log U)$-time, deterministic algorithm for solving the single-source shortest paths problem in a directed graph G in which all edge weights are in $[U]$. For reference, in the undirected case, there is an $O(m + n)$-time algorithm due to Thorup for shortest paths assuming that each integer fits into a machine word.