6.851: ADVANCED DATA STRUCTURES, SPRING 2021
Prof. Erik Demaine, Josh Brunner, Dylan Hendrickson, Yevhenii Diomidov

Problem Set 7 Solutions

Due: Thursday, April 8, 2021

Problem 7.1 [Van Emde Boas in Space].

Solution:

(a) We prove matching bounds of $\Theta(n \log\log u)$.

Upper bound: An insert operation takes $O(\log\log u)$ time, and thus increases the size of the data structure by $O(\log\log u)$. We can build a vEB structure with any set of n elements using n insert operations, so the size is $O(n \log\log u)$.

Lower bound: For sufficiently large n, let $2^{\log^c n} \le u \le 2^{2^{dn}}$ for some constants $c > 1$ and d. Consider a set with n elements equally spaced throughout u. These elements are in different clusters, and will continue to be in different clusters as long as possible: until the size of the universe is less than n^2. Since they are in different clusters, other than one which is stored as the min, they all incur $\Theta(1)$ space in their clusters and are recorded in the summary structure. This continues recursively: at the level-i summary structure, there are $n - i$ remaining elements which each incur $\Theta(1)$ space, and the size of the universe is $u^{1/2^i}$. This continues as long as the size of the universe is at least n^2, meaning

$$u^{1/2^i} \ge n^2 \iff \frac{1}{2^i} \log u \ge 2 \log n \iff \frac{\log u}{\log n} \ge 2^{i+1} \iff \log\log u - \log\log n - 1 \ge i.$$

We chose u such that $\log\log u \ge c \log\log n$ and $c > 1$, so $\log\log u - \log\log n - 1 = \Omega(\log\log u)$. In particular, there are $\Omega(\log\log u)$ levels of recursion, each of which uses $\Theta(n - i)$ space. So the total space is $\Omega(\min\{n \log\log u, n^2\})$. We chose u such that $\log\log u = O(n)$, so $\min\{n \log\log u, n^2\} = \Omega(n \log\log u)$; thus the total space is $\Omega(n \log\log u)$.

Other possible answers for $f(n, u)$ include $n \log\log \frac{n}{u}$, n^2, and u.

(b) Use indirection: store each group of $\Theta(\log u)$ consecutive elements in the set in a balanced binary search tree, with only one element from each group in the vEB structure. The vEB structure stores $O(\frac{n}{\log u})$ elements, so it takes space $O(\frac{n \log\log u}{\log u}) = o(n)$, and the total space of the binary trees is $O(n)$. In each operation, we search the vEB structure in $O(\log\log u)$ time, and then search the relevant binary tree in $O(\log\log u)$ time. When a tree has size less than $\frac{1}{2} \log u$ or more than $2 \log u$, merge it with an adjacent tree or split it in $O(\log\log u)$ time.

One can alternatively use groups of $\Theta(\log\log u)$ elements, stored in a binary tree or even just as a list, taking $O(\log\log u)$ time to traverse.