6.851: Advanced Data Structures, Fall 2017

Prof. Erik Demaine, Adam Hesterberg, Jayson Lynch

Problem Set 5 Solutions

Due: Wednesday, October 11, 2017 at noon

Problem 5.1 [Ordered File Maintenance without Extra Space].

Describe a data structure for inserting n elements into an initially empty array of length exactly n in $O(\lg^3 n)$ amortized swaps per operation.

That is, you're given n spots for items. Initially, each spot is empty. One at a time, you're given an item to insert into a spot and an order relative to the existing items; you may swap other items' positions if necessary. You want to perform all n inserts using $O(n \lg^3 n)$ total swaps.

Solution:

Partition the n insertions into groups of sizes $\frac{n}{2}$, $\frac{n}{4}$, ..., 2, 1. We'll spend $O(n \lg^2 n)$ swaps on each group, for a total of $O(n \lg^3 n)$.

Before inserting the group of size 2^i, we'll read the entire array and space the already-inserted elements out perfectly (into blocks of size $2^{\lg n - i}$ consecutive items), which takes $O(n)$ swaps for each group, or $O(n \lg n)$ swaps overall.

To process the group of size 2^i, pretend first that we could ignore the already-inserted items and do ordered file maintenance on the 2^{i+1} empty slots available. As covered in class, that takes $O(\lg^2(2^i)) < c \lg^2 n$ amortized swaps per operation, or a total of $c2^i \lg^2 n$ swaps for the group.

Since there are already-inserted items, when we add an item to an "empty" slot in the pretended problem, we might have to move up to $2^{\lg n - i}$ consecutive items; when we swap items in the pretended problem, we have to swap $2^{\lg n - i}$ items. So, each of the $c2^i \lg^2 n$ swaps in the pretend problem requires at most $2^{\lg n - i}$ swaps to implement, for a total of at most $cn \lg^2 n$ swaps for the whole group, or $cn \lg^3 n$ swaps for all $\lg n$ groups, as desired.