University of Bristol
COMS21103: Data Structures and Algorithms
Problem Set 7

Problem 1: A sequence of n operations is performed on a data structure. The ith operation costs i if i is an exact power of 2, and 1 otherwise. Use aggregate analysis to determine the amortised per operation.

Problem 2: A sequence of stack operations is performed on a stack whose size never exceeds k. After every k operations, a copy of the entire stack is made for backup purposes. Show that the cost of n stack operations, including copying the stack, is $O(n)$ by assigning suitable amortised costs to the various stack operations.

Problem 3: Execute the following operations on an initially empty Fibonacci heap:

Insert(18), Insert(14), Insert(17), Insert(28), Insert(32), Insert(37),
Insert(25), Insert(36), Insert(53), Insert(40), ExtractMin(), DecreaseKey(40, 30),
Delete(36), ExtractMin().

For all intermediate steps, illustrate the resulting Fibonacci heap. New elements should always be inserted to the right of the current minimum. The consolidation operation after ExtractMin starts with the next element on the right hand side of the deleted minimum.

Problem 4: Professor Pinocchio claims that the height of an n-node Fibonacci heap is $O(\lg n)$. Show that the professor is mistaken by exhibiting, for any positive integer n, a sequence of Fibonacci-heap operations that creates a Fibonacci heap consisting of just one tree that is a linear chain of n nodes.

Problem 5: What is the *actual* (not amortised) worst-case running time of a single DecreaseKey operation on a Fibonacci heap with n elements (in O-notation)? Justify your answer.

Problem 6: Suppose that we generalise the cascading-cut rule to cut a node x from its parents as soon as it loses its kth child, for some integer k. Prove that, as long as $k = O(1)$, specific choices of k do not change the asymptotic amortised costs of the operations discussed in class.