

---

## Efficient Algorithms and Datastructures I

---

### Question 1 (10 Points)

Show that any sequence of  $m$  MAKE-SET, FIND-SET, and LINK operations, where all the LINK operations appear before any of the FIND-SET operations, takes only  $O(m)$  time if both path compression and union by rank are used. What happens in the same situation if only the path-compression heuristic is used?

### Question 2 (10 Points)

In cuckoo hashing, we double or halve the hash table size and rehash all the elements to ensure that the fill-factor is within a range. Explain formally how we can amortize this cost for rehashing against cost for insertions and deletions, by setting up a potential function.

### Question 3 (10 Points)

Suppose instead of using decimal or dual representation of numbers, we represent them in binary over the basis of Fibonacci numbers. That is, the bit-string  $(X_k, X_{k-1}, \dots, X_1)_F$  represents the number  $n = \sum_{i=1}^k X_i \cdot F_i$ , where  $F_i$  denotes the  $i$ th Fibonacci number ( $F_1 = F_2 = 1$  and  $F_i = F_{i-1} + F_{i-2}$  for  $i \geq 3$ ). For example,  $(31)_{10}$  can be represented by the bit string  $(10100100)_F$  since  $F_8 + F_6 + F_3 = (21)_{10} + (8)_{10} + (2)_{10} = (31)_{10}$ , and also by the bit string  $(10011011)_F$  since  $F_8 + F_5 + F_4 + F_2 + F_1 = (21)_{10} + (5)_{10} + (3)_{10} + (1)_{10} + (1)_{10} = (31)_{10}$ .

- Argue that we can represent any number  $n \in \mathbb{N}_0$  like this.
- Describe an algorithm which performs increment and decrement operations in this representation in constant amortized time (starting from 0). Assume that flipping each bit requires one unit of work.  
(*Hint*: Use a potential function that assigns potential depending on whether consecutive pairs of bits are similar. For example, if bit  $i$  and bit  $i + 1$  are equal/different, you may say they contribute one unit to the potential. Make sure that the potential of  $(0)_F$  is zero units.)