Analysis of Fibonacci Array Growth

The Fibonacci sequence is defined by

$$F_{n+2} = F_{n+1} + F_n \quad \text{for } n \geq 0,$$

with initial conditions $F_1 = F_2 = 1$.

Subtask I

For all $n \geq 1$,

$$1 + F_1 + F_2 + \cdots + F_n = F_{n+2}.$$

Proof. Prove by mathematical induction.

Base case: Let $n = 1$.

$$1 + F_1 = 1 + 1 = 2,$$

$F_3 = F_2 + F_1 = 1 + 1 = 2$

Inductive hypothesis: Assume that for some $n \geq 1$,

$$1 + \sum_{k=1}^{n} F_k = F_{n+2}.$$

Inductive step: show that

$$1 + \sum_{k=1}^{n+1} F_k = F_{n+3}.$$

Using the inductive hypothesis,

$$1 + \sum_{k=1}^{n+1} F_k = \left(1 + \sum_{k=1}^{n} F_k\right) + F_{n+1} = F_{n+2} + F_{n+1}.$$

By the Fibonacci recurrence relation,

$$F_{n+3} = F_{n+2} + F_{n+1}.$$

so,

$$1 + \sum_{k=1}^{n+1} F_k = F_{n+3}.$$

Therefore, the statement holds for all $n \geq 1$. $\square$

Subtask II

For all $n \geq 1$,

$$\frac{1}{F_n}\left(1 + \sum_{k=1}^{n} F_k\right) \le 3.$$

Proof. By Theorem 1,

$$1 + \sum_{k=1}^{n} F_k = F_{n+2}.$$

Therefore,

$$\frac{1}{F_n}\left(1 + \sum_{k=1}^{n} F_k\right) = \frac{F_{n+2}}{F_n}.$$

Using the Fibonacci recurrence,

$$F_{n+2} = F_{n+1} + F_n = (F_n + F_{n-1}) + F_n = 2F_n + F_{n-1}.$$

so,

$$\frac{F_{n+2}}{F_n} = 2 + \frac{F_{n-1}}{F_n}.$$

Since the Fibonacci sequence is increasing, we have

$$\frac{F_{n-1}}{F_n} \le 1.$$

Hence,

$$\frac{F_{n+2}}{F_n} \le 3.$$

□

Subtask III

We analyze the total number of copy operations when using the Fibonacci growth strategy for an ArrayList.

Initially, the array has capacity $F_2 = 1$. Whenever the array becomes full with capacity F_k, it is resized to capacity F_{k+1}, requiring F_k elements to be copied.

Assume that $n = F_r + 1$ for some integer $r \geq 2$. Then resizes occur for capacities

$$F_2, F_3, \dots, F_r.$$

The total number of copy steps is therefore

$$\sum_{k=2}^{r} F_k.$$

By Theorem 1,

$$\sum_{k=1}^{r} F_k = F_{r+2} - 1,$$

which implies

$$\sum_{k=2}^{r} F_k \le F_{r+2} - 1.$$

From Subtask II, we know that

$$F_{r+2} \leq 3F_r.$$

Since $n = F_r + 1$, it follows that $F_r \leq n$. Therefore,

$$\sum_{k=2}^{r} F_k = O(n).$$

Thus, inserting n elements using the Fibonacci growth scheme requires a total of $O(n)$ copy operations, yielding an amortized cost of $O(1)$ per insertion. 0□