Texas A&M University

Research Project 1

Some Conjectures on Fibonacci Numbers

Authors:
Daniel Whatley
Sarah Sahibzada
Taylor Wilson

Supervisor:
Dr. Sara Pollock

October 23, 2015

Contents

1 Introduction

The *Fibonacci sequence*, $\{F_n\}$, is defined as follows:

$$F_1 = 1, \qquad F_2 = 1, \qquad F_n = F_{n-1} + F_{n-2} \quad \text{for} \quad n \geq 3,$$

with seed values $F_0 = 0$ and $F_1 = 1$. Similarly, the Lucas numbers are a sequence of integers defined by the following recurrence:

$$L_n = L_{n-1} + L_{n-2}$$

with seed values $L_1 = 1$ and $L_2 = 3$; alternative definitions begin this recurrence as $L_0 = 2$ and $L_1 = 1$. In this paper we analyze two conjectures: one about which Fibonacci numbers are perfect squares, and on various identities linking the Fibonacci sequence and the Lucas numbers.

These numbers are known to be closely related to the Fibonacci sequence. This was investigated computationally.

A number of identities were verified computationally up to the first 175 Lucas and Fibonacci numbers as well.

2 Theoretical Analysis

2.1 Fibonacci Squares

It was a long-standing conjecture, until it was proved in 1964 [2], that the only Fibonacci numbers that are perfect squares are $F_1 = 1$, $F_2 = 1$, and $F_{12} = 144$. M. Wunderlich[1] described an ingenious “exclusion” method to calculate which Fibonacci numbers, out of the first million, can possibly be perfect squares. The method does not prove that any Fibonacci numbers are perfect squares, rather it rules out those that are not.

In Section 3, we describe some limitations of this approach, and describe an optimization.

2.2 Lucas Numbers and Associated Identities

Insofar as the Fibonacci and Lucas sequences, and their sums, may be developed in different ways with various identities and recurrences, $F_{n+k} + F_{n-k} = F_n L_k$, even k
$F_{n+k} + F_{n-k} = L_n F_k$, odd k
$F_{n+k} - F_{n-k} = F_n L_k$, odd k
$F_{n+k} - F_{n-k} = L_n F_k$, even k

$L_{n+k} + L_{n-k} = L_n L_k$, even k
$L_{n+k} - L_{n-k} = F5_n F_k$, even k
$L_{n+k} + L_{n-k} = F_n L_k$, even k
It is possible to obtain individual Lucas numbers from Fibonacci numbers using various identities relating these quantities:
$L_n = F_{n-1} + F_{n+1}$

$$\begin{bmatrix} L_{n+1} \\ L_n \end{bmatrix} = Q_L \begin{bmatrix} F_n \\ F_{n-1} \end{bmatrix}$$

as well as

$$5 \begin{bmatrix} F_{n+1} \\ L_n \end{bmatrix} = Q_L \begin{bmatrix} L_n \\ L_{n-1} \end{bmatrix}$$

, where we define $Q_L = \begin{bmatrix} 3 & 1 \\ 1 & 2 \end{bmatrix}$

3 Computational Approach

3.1 Fibonacci Squares

[1] found that the only Fibonacci numbers up to $F_{1000000}$ are the three described above using a computational approach. The researchers here took a similar approach, but used optimization techniques to decrease running time and to give more conclusive results. The programming language used in this research was predominantly C++.

Consider a prime p. We first calculate the period of the Fibonacci numbers $\pmod p$, denoted here as the *Fibonacci period* of p. For example, the Fibonacci period of $p = 7$ is 16, as the Fibonacci sequence $\pmod 7$ repeats after 16 steps. Specifically, the first 16 Fibonacci numbers $\pmod 7$ are:

$$1, 1, 2, 3, 5, 1, 6, 0, 6, 6, 5, 4, 2, 6, 1, 0,$$

after which it repeats $1, 1, 2, 3, 5, \ldots$ again. Because the numbers $\pmod 7$ that are quadratic non-residues are 3, 5, and 6, replacing each quadratic residue by a 1 and each quadratic non-residue by a 0 gives the binary string

$$111001010001101.$$

Let this string for each prime p be S_p.

Suppose we take a list of primes: $p_1, p_2, \ldots, p_n$, and let the Fibonacci periods of each prime be $d_1, d_2, \ldots, d_n$. Take those indices of S_{p_i} that are 0.

If any Fibonacci number is congruent to one of these indices mod d_i, then it can never be a perfect square of any integer. Such Fibonacci numbers can now be eliminated due to this prime. Repeat for more primes, until the number of possible square Fibonacci numbers is down to a reasonable number to analyze. We expect that approximately half of the Fibonacci numbers will be eliminated at each step, as about half of the integers mod a prime p are quadratic residues. If $L = \mathrm{lcm}(d_1, d_2, \ldots, d_n)$, the worst case running time of this algorithm is $O(L \log n)$, where n is the number of primes considered.

The computation in [1] was somewhat limited for two reasons. First, for each prime p, the quadratic residues and non-residues mod p had to be reiterated all the way to the computation limit (in the paper, the computation limit is 10^6). This means that a loop of size 10^6 had to be performed each time a new prime was considered. The computation here is superior for another reason: any F_k with $k \not\equiv a \pmod{L}$ where F_a is a perfect square is therefore also not a perfect square. This is because if $k \not\equiv a \pmod{L}$, then there exists some d_i for which $k \not\equiv a \ (\mathrm{mod}\ d)_i$, which also means that F_k is not a quadratic residue modulo p_i, which means F_k cannot be a perfect square. Because 10^6 is not perfectly divisible by each of the Fibonacci periods considered, the computation cannot easily be extended to all positive integers, but the computation here can. This is how the approach was optimized.

3.2 Lucas Numbers and Various Identities

Large lists of Fibonacci and Lucas numbers were generated using a Python script that utilized (matrix identity for fibonacci) in order to generate Fibonacci numbers, and generated Lucas numbers using the Fibonacci sequence.

4 Results and Conclusion

Since the value of L cannot be too large for our purposes (the computational size is linear in L), we must find primes with relatively small Fibonacci periods. To do this, we limited our primes p_i to below 10^5, and gave limits on $v_2(d_i)$, $v_3(d_i)$, $v_5(d_i)$, $v_7(d_i)$, $v_9(d_i)$, $v_{11}(d_i)$ where $v_q(d_i)$ is the power of prime q in the prime factorization of d_i. Different files were generated based on these limits, each specifying a value of L. After this algorithm was run on each file, the possible residues $r_1, r_2, \ldots \pmod{L}$ such that $F_{r_1}, F_{r_2}, \ldots$ can be perfect squares were outputted.

In each run, we got that the only possible Fibonacci numbers that can be perfect squares were the ones with indices $1, 2, 12 \pmod{L}$, confirming the computation done in [1] and the proof done in [2]. Running times on a supercomputer are summarized as follows:

L	Number of primes	Execution time (s)
604800	103	0.88
907200	112	1.34
6350400	138	9.24
9072000	135	12.37
63504000	163	105.61

5 Individual Contributions

1. Sarah: Wrote code to verify Fibonacci and Lucas sum identities; wrote small scripts to generate Lucas and Fibonacci numbers based on known identities; implemented the Fibonacci squares algorithm in parallel in C; re-implemented the Fibonacci and Lucas sum code in parallel. Developed some theoretical background on the Lucas sequence.

2. Daniel: I mainly wrote code for verifying the Fibonacci squares conjecture. I implemented an algorithm to find relatively reasonable Fibonacci periods, generated different files for verification, and implemented various algorithms to find quadratic residues and non-residues, powers of integers modulo others, and other elementary number theory constructs required for thorough analysis of the subject. I also developed theoretical background in this conjecture and for the optimization of the computation in [1]. Finally, I collected results for this conjecture and produced some tables.

3. Taylor:

6 References

1. M. Wunderlich, On the non-existence of Fibonacci Squares, Maths, of Computation, 17 (1963) p. 455.

2. J. H. E. Cohn, “Square Fibonacci Numbers, Etc.” Fibonacci Quarterly **2** 1964, pp. 109-113.

3. Other stuff about other conjectures here