On the algorithms of Fibonacci sequence

Chanyut Yuvacharuskul

September 30, 2021

Abstract

This paper illustrates the inner workings of a GitHub repository of `fibonacci` by the username `ChanyutJean` that outputs a Fibonacci number given an index. The details are proof-oriented, and may not divulge in too much technical details surrounding system limitations like integer size and floating point precision.

1 Introduction

There are many algorithms to find a Fibonacci number; some are exponentially slow, and some are very efficient. The GitHub repository found in `https://github.com/ChanyutJean/fibonacci` presents ways to output a Fibonacci number when given an index. We will analyze them in this paper, and also present supporting proofs to accompany the explanation later in this paper.

2 Explanation

2.1 Perl

The Fibonacci sequence is a recurrence relation $F_n = F_{n-1} + F_{n-2}$ satisfying base case $F_1 = 1$ and $F_2 = 1$. The first few numbers in the sequence are $1, 1, 2, 3, 5, 8, 13,$ and 21. It is a deceptively simple yet meaningful sequence that can be calculated in many ways, as we shall see in this section.

The Fibonacci sequence can be illustrated by a rabbit family example, implemented in the Perl file. Suppose that a rabbit produces an offspring every year after 2 years it is born. If at the start there are only one rabbit, then for the first two years there will only be one rabbit, and for the rest of the years the number of rabbits will be the ones who continued living from last year plus the newborn rabbit amounting to the ones from the last two years. The recurrence relation exhibited in this family will thus be

$$F_n = F_{n-1} + F_{n-2}$$

with base cases $F_1 = F_2 = 1$.

2.2 Haskell

The Fibonacci sequence defined in Haskell is `f[1] = 1; f[2] = 1; f[3+] = f[1+] + f[2+]`. In the first and second element of the array, and value is 1. The rest of the elements come from the addition of the array shifted once with itself shifted twice (both to the left). So, the third element will be `f[1] + f[2]`, and the fourth element will be `f[2] + f[3]`, satisfying the Fibonacci recurrence relation.

2.3 Python

In the Python implementation, a naively recursive function is used to determine the values. The recursion ends with the base cases $F_1 = F_2 = 1$.

2.4 JavaScript

In the JavaScript implementation, in finding the number F_n, $n+1$ elements in an array starting from $F_0 = 0$ and the base cases are utilized to iteratively find the next number in the sequence until the n^{th} number is found, taking $O(n)$ in both time and space complexity to complete.

2.5 Bash

The Bash implementation reduces the space complexity to $O(1)$, holding only the two latest numbers to iterate. This is possible since the Fibonacci sequence only relies on the last two numbers in the sequence.

2.6 PHP

PHP's code infinitesimally reduces the time taken to find F_n by starting with 2 as the current index, storing $F_1 = F_2 = 1$ instead of $F_0 = 0, F_1 = 1$. If the index of 1 is requested, F_2 is instead returned, with coincidentally the same value as F_1.

2.7 Go

In Go, more than one variables can be assigned at the same time, and the implementation stores the next value in the previous index and then swapping the previous index with the current index. This reduces the variable size to 2.

2.8 Lua

Another way to find the Fibonacci numbers is from the identity (see proof in Section 3.1):

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^n = \begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix}$$

The implementation multiplies the matrix on the left-hand side by itself $n-1$ times and returns the top left value of the right-hand side matrix.

2.9 Java

The Java implementation makes use of the preceding identity, but with fast exponentiation. In fast exponentiation, the identity is:

$$x^n = \begin{cases} (x^2)^{\frac{n}{2}} & n \text{ is even} \\ x(x^2)^{\frac{n-1}{2}} & n > 1 \text{ is odd} \\ x & n = 1 \end{cases}$$

2.10 CommonLisp

CommonLisp is proficient in computing math equations, and the Fibonacci sequence can be determined as (see proof in Section 3.2)

$$F_n = \frac{\phi^n - \psi^n}{\phi - \psi}$$

where $\phi = \frac{1+\sqrt{5}}{2}$ and $\psi = \frac{1-\sqrt{5}}{2}$.

2.11 Rust

The sequence of Fibonacci hides in Pascal's Triangle; it is the sum of the n^{th} diagonal slanted up starting at the top going downwards. To illustrate, the first few lines of the Pascal Triangle are:

$$\begin{array}{ccccccccc}
 & & & & \binom{0}{0} & & & & \\
 & & & \binom{1}{0} & & \binom{1}{1} & & & \\
 & & \binom{2}{0} & & \binom{2}{1} & & \binom{2}{2} & & \\
 & \binom{3}{0} & & \binom{3}{1} & & \binom{3}{2} & & \binom{3}{3} & \\
\binom{4}{0} & & \binom{4}{1} & & \binom{4}{2} & & \binom{4}{3} & & \binom{4}{4}
\end{array}$$

and the first few Fibonacci numbers are:

$$\begin{aligned}
F_1 &= 1 = \binom{0}{0} \\
F_2 &= 1 = \binom{1}{0} \\
F_3 &= 2 = \binom{2}{0} + \binom{1}{1} \\
F_4 &= 3 = \binom{3}{0} + \binom{2}{1} \\
F_5 &= 5 = \binom{4}{0} + \binom{3}{1} + \binom{2}{2}
\end{aligned}$$

With this, one can now deduce (with proof on Section 3.3) the following:

$$F_n = \sum_{k=0}^{\lfloor \frac{n-1}{2} \rfloor} \binom{n-k-1}{k}$$

and that is the implementation deployed in Rust.

2.12 Ruby

Lastly, we will show the Ruby implementation of the Fibonacci sequence. The Fibonacci sequence can be shown to be equal to the number of ways 1's and 2's and be added together sequentially to obtain the number lower than the index by one. For example, reminding that $F_5 = 5$, there are 5 ways to obtain the number $5 - 1 = 4$, which are

$$\begin{aligned}
4 &= 1 + 1 + 1 + 1 \\
&= 1 + 1 + 2 \\
&= 1 + 2 + 1 \\
&= 2 + 1 + 1 \\
&= 2 + 2
\end{aligned}$$

To transition to the proof section, we shall give an elementary proof that the implementation in Ruby actually gives a Fibonacci sequence. First, notice that there can only be one way to represent zero, which is adding nothing. Furthermore, there is also only one way to represent one, which is the number itself. This aligns with $F_1 = F_2 = 1$. Now, the ways to get n by adding only 1's and 2's is to add 1 to the ways to get $n - 1$ and 2 to the ways to get $n - 2$. This gives the formula $F_n = F_{n-1} + F_{n-2}$, which completes the intuition.

3 Proof

3.1 Lua: Matrix Implementation

We will now prove the following identity by induction:

$$\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^n = \begin{pmatrix} F_{n+1} & F_n \\ F_n & F_{n-1} \end{pmatrix}$$

As for base case, it is easy to see that $n = 1$ gives the correct Fibonacci numbers of F_0, F_1, and F_2, given we define $F_0 = 0$. Now we will assume that the identity is true for $1 \leq m \leq n - 1$. This gives:

$$\begin{aligned}
\begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^n &= \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}^{n-1} \times \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \\
&= \begin{pmatrix} F_n & F_{n-1} \\ F_{n-1} & F_{n-2} \end{pmatrix} \times \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} \\
&= \begin{pmatrix} F_n + F_{n-1} & F_n \\ F_{n-1} + F_{n-2} & F_{n-1} \end{pmatrix} \\
&= \begin{pmatrix} F_{n+1} & F_n \\ F_n + F_{n-1} & F_{n-1} \end{pmatrix}
\end{aligned}$$

and this proves the identity.

3.2 CommonLisp: Binet's Formula

We now proof by induction Binet's formula that calculates a Fibonacci number given it's index. The formula states:

$$F_n = \frac{\phi^n - \psi^n}{\phi - \psi}$$

where $\phi = \frac{1+\sqrt{5}}{2}$ and $\psi = \frac{1-\sqrt{5}}{2}$. The base case, $n = 1$, is verified by $F_1 = \frac{\phi^1 - \psi^1}{\phi - \psi} = 1$. We then assume the formula for all $1 \leq m \leq n - 1$ for the inductive case, and see the following:

$$\begin{aligned}
F_n &= F_{n-1} + F_{n-2} \\
&= \frac{\phi^{n-1} - \psi^{n-1}}{\phi - \psi} + \frac{\phi^{n-2} - \psi^{n-2}}{\phi - \psi} \\
&= \frac{\phi^{n-2}(\phi + 1) - \psi^{n-2}(\psi + 1)}{\phi - \psi} \\
&= \frac{\phi^{n-2}\phi^2 - \psi^{n-2}\psi^2}{\phi - \psi} \\
&= \frac{\phi^n - \psi^n}{\phi - \psi}
\end{aligned}$$

where we have used:

$$\phi^2 = \left(\frac{1 + \sqrt{5}}{2}\right)^2 = \frac{6 + 2\sqrt{5}}{4} = \frac{1 + \sqrt{5}}{2} + 1 = \phi + 1$$

and similarly, $\psi^2 = \psi + 1$, completing the proof.

3.3 Rust: Combinatorial Formula

We wish to proof the following by induction:

$$F_n = \sum_{k=0}^{\lfloor \frac{n-1}{2} \rfloor} \binom{n-k-1}{k}$$

Before we begin, we will proof a lemma combinatorially. The lemma is:

$$\binom{n-1}{k-1} + \binom{n-1}{k} = \binom{n}{k}$$

Both sides count ways to obtain k red balls in a total of n balls. The left hand side correspond to two cases to count: $\binom{n-1}{k-1}$ are ways to obtain the rest of $k-1$ red balls in $n-1$ balls to combine with another red ball that is picked beforehand, and $\binom{n-1}{k}$ are ways to obtain k red balls in $n-1$ balls where a non-red ball is picked beforehand.

To begin, first note that $F_1 = \sum_{k=0}^{\lfloor \frac{1-1}{2} \rfloor} \binom{1-k-1}{k} = \binom{0}{0} = 1$ and $F_2 = \sum_{k=0}^{\lfloor \frac{2-1}{2} \rfloor} \binom{2-k-1}{k} = \binom{1}{0} = 1$. Now we assume the formula for $1 \leq m \leq n-1$ and see the following:

$$\begin{aligned} F_n &= F_{n-1} + F_{n-2} \\ &= \sum_{k=0}^{\lfloor \frac{n-2}{2} \rfloor} \binom{n-k-2}{k} + \sum_{k=0}^{\lfloor \frac{n-3}{2} \rfloor} \binom{n-k-3}{k} \end{aligned}$$

Noting that $\lfloor n/2 \rfloor$ equals $n/2$ for even n and $(n-1)/2$ for odd n, we now split the cases between odd n and even n. For odd n:

$$\begin{aligned} F_n &= \sum_{k=0}^{\frac{n-3}{2}} \binom{n-k-2}{k} + \sum_{k=0}^{\frac{n-3}{2}} \binom{n-k-3}{k} \\ &= \sum_{k=1}^{\frac{n-3}{2}} \binom{n-k-2}{k} + \sum_{k=0}^{\frac{n-5}{2}} \binom{n-k-3}{k} + \binom{n-k-2}{k}\bigg|_{k=0} + \binom{n-k-3}{k}\bigg|_{k=\frac{n-3}{2}} \\ &= \sum_{k=0}^{\frac{n-5}{2}} \binom{n-k-3}{k+1} + \sum_{k=0}^{\frac{n-5}{2}} \binom{n-k-3}{k} + \binom{n-3}{0} + \binom{n-\frac{n-3}{2}-3}{\frac{n-3}{2}} + \\ &= \sum_{k=0}^{\frac{n-5}{2}} \left[\binom{n-k-3}{k+1} + \binom{n-k-3}{k} \right] + \binom{n-3}{0} + \binom{\frac{n-3}{2}}{\frac{n-3}{2}} \\ &= \sum_{k=0}^{\frac{n-5}{2}} \binom{n-k-2}{k+1} + 1 + 1 \text{ (by the lemma)} \\ &= \sum_{k=1}^{\frac{n-3}{2}} \binom{n-k-1}{k} + \binom{n-0-1}{0} + \binom{n-\frac{n-1}{2}-1}{\frac{n-1}{2}} \\ &= \sum_{k=0}^{\frac{n-1}{2}} \binom{n-k-1}{k} \\ &= \sum_{k=0}^{\lfloor \frac{n-1}{2} \rfloor} \binom{n-k-1}{k} \end{aligned}$$

Finally, the formula will be proven once the case for even n is shown:

$$\begin{aligned}
F_n &= \sum_{k=0}^{\frac{n-2}{2}} \binom{n-k-2}{k} + \sum_{k=0}^{\frac{n-4}{2}} \binom{n-k-3}{k} \\
&= \sum_{k=1}^{\frac{n-2}{2}} \binom{n-k-2}{k} + \sum_{k=0}^{\frac{n-4}{2}} \binom{n-k-3}{k} + \binom{n-k-2}{k}\bigg|_{k=0} \\
&= \sum_{k=0}^{\frac{n-4}{2}} \binom{n-k-3}{k+1} + \sum_{k=0}^{\frac{n-4}{2}} \binom{n-k-3}{k} + \binom{n-2}{0} \\
&= \sum_{k=0}^{\frac{n-4}{2}} \left[\binom{n-k-3}{k+1} + \binom{n-k-3}{k}\right] + 1 \\
&= \sum_{k=0}^{\frac{n-4}{2}} \binom{n-k-2}{k+1} + 1 \\
&= \sum_{k=1}^{\frac{n-2}{2}} \binom{n-k-1}{k} + \binom{n-0-1}{0} \\
&= \sum_{k=0}^{\frac{n-2}{2}} \binom{n-k-1}{k} \\
&= \sum_{k=0}^{\lfloor \frac{n-1}{2} \rfloor} \binom{n-k-1}{k}
\end{aligned}$$

and this concludes the proofs of the presented equation.

4 Conclusion

Now, the inner workings of the Fibonacci repository is shown in full-view. The Fibonacci sequence is a curious sequence; it has brought upon all the magnificent depiction of the beauty of mathematics. The author hopes the delve into the many facets of the Fibonacci sequence has been somewhat eye-opening.