Arithmetic of Fibonacci representations

I.S. Sergeev, 2021

Fibonacci codes: $\ldots 0100101010000100000001010100010000101 0000 \ldots$

no '11' subwords

The number of length-n words ... $\{\bigstar^n\} = \{0\bigstar^{n-1}\} \cup \{10\bigstar^{n-2}\}$

... is Φ_{n+2} (Φ_k — Fibonacci numbers: $\Phi_1 = \Phi_2 = 1,\ \Phi_k = \Phi_{k-1} + \Phi_{k-2}$).

$0 = 0000,\ 1 = 0001,\ 2 = 0010,\ 3 = 0100,\ 4 = 0101,\ 5 = 1000,\ 6 = 1001,\ 7 = 1010$

Fibonacci (canonical) representation of a number A: $[a_n, \ldots, a_3, a_2]$:

$$A = \sum_{k=2}^{n} a_k \Phi_k, \qquad a_k \in \{0,\ 1\}, \qquad a_{k+1} a_k = 0.$$

Lines of research:

(i) arithmetic operations in Fibonacci encoding $(+, -, *)$

(ii) transitions between the Fibonacci and the binary representations of a number

Addition (subtraction) of Fibonacci encodings:

$$
\begin{array}{cr}
 & 10101010 \\
+ & 10100101 \\
\hline
= & \underline{20}201111 \\
 & 10\underline{030}1111 \\
 & 101\underline{102}111 \\
 & 1\underline{011}11011 \\
 & 110\underline{011}011 \\
 & 110100\underline{011} \\
 & \underline{11}0100100 \\
 & \boxed{1000100100}
\end{array}
$$

Rules:

$$
\begin{array}{rcl}
020x & \to & 100x' \\
030x & \to & 110x' \\
021x & \to & 110x \\
012x & \to & 101x \\
 & & \\
011x & \to & 100x
\end{array}
$$

Complexity $O(n)$ [Berstel, Frougny, Ahlbach, Usatine, Pippenger ≈1986–2011]

Multiplication (???)

in binary encoding: $M(n) = O(n \log n)$ [Harvey, van der Hoeven 2019]

Transition to binary encoding: $O(n^2)$ [Ahlbach, Usatine, Pippenger 2011]

Schönhage's method: $A = [a_{k-1}, \ldots, a_0]_b \longrightarrow A = [a'_{n-1}, \ldots, a'_0]_2$

$A = [A_1]_b \cdot b^{k/2} + [A_0]_b$

$\downarrow$

$A = [A_1]_2 \cdot [b^{k/2}]_2 + [A_0]_2$

Complexity:

$$T(n) \le 2T(n/2) + M(n/2) + O(n)$$

$$\Longrightarrow T(n) = O(M(n) \log n)$$

Lucas numbers: $L_m = \Phi_{m+1} + \Phi_{m-1} = \varphi_+^m + \varphi_-^m$, where $\varphi_\pm = \frac{1 \pm \sqrt{5}}{2}$

$$\Phi_k L_m = \Phi_{m+k} - (-1)^k \Phi_{m-k} \qquad (\star)$$

Algorithm: $A = [a_{n-1}, \ldots, a_0]_2 \longrightarrow A = [a'_{k-1}, \ldots, a'_0]_\varphi$

$A = A_1 \cdot L_m + A_0$ division with remainder by $L_m \asymp 2^{n/2}$

$\downarrow$

$$A = [A_1]_\varphi \cdot L_m + [A_0]_\varphi \overset{(\star)}{=} [A^+]_\varphi - [A^-]_\varphi + [A_0]_\varphi$$

Complexity: $T(n) \le 2T(n/2) + D(n) + O(n) \Rightarrow T(n) = O(M(n) \log n)$

r-Fibonacci representations: <u>no r ones in a row</u>

r-bonacci numbers: $\Phi_n^{(r)} = \Phi_{n-1}^{(r)} + \ldots + \Phi_{n-r}^{(r)}$

$$A = \sum_{k=2}^{n} a_k \Phi_k^{(r)}, \qquad a_k \in \{0,\, 1\}, \qquad a_{k+1} \cdot \ldots \cdot a_{k+r} = 0.$$

$\langle q \rangle$-Fibonacci representations: <u>ones are separated by $\geq q-1$ zeros</u>

$\langle q \rangle$-bonacci numbers: $\Phi_n^{\langle q \rangle} = \Phi_{n-1}^{\langle q \rangle} + \Phi_{n-q}^{\langle q \rangle}$

$$A = \sum_{k=2}^{n} a_k \Phi_k^{\langle q \rangle}, \qquad a_k \in \{0,\, 1\}, \qquad a_{k+1} + \ldots + a_{k+q} \leq 1.$$

---

transition $[A]_2 \longleftrightarrow [A]_{\varphi(r)}$, $[A]_2 \longleftrightarrow [A]_{\varphi\langle 3 \rangle}$: $T(n) = 2^{O(\sqrt{\log n})} n.$

Addition of Tribonacci encodings: $O(n)$.

Addition/subtraction in other cases: (???).

References:

Ahlbach C., Usatine J., Frougny C., Pippenger N. Efficient algorithms for Zeckendorf arithmetic // Fibonacci Quarterly. 2013. 51(3), 249–255.

Sergeev I.S. On the complexity of Fibonacci coding // Problems of Information Transmission. 2018. 54(4), 343–350.