Project Euler #684: Inverse Digit Sum

Solved 29 November, 2021

Define $s(n)$ to be the smallest number that has a digit sum of n. For example $s(10) = 19$.

Let $S(k) = \sum_{n=1}^{k}$. You are given $S(20) = 1074$.

Further let f_i be the Fibonacci sequence defined by $f_0 = 0, f_1 = 1$ and $f_i = f_{i-2} + f_{i-1}$ for all $i \geq 2$.

Find $\sum_{i=2}^{90} S(f_i)$. Give your answer modulo $1\ 000\ 000\ 007$.

Analysis

First, we notice that to generate $s(n)$, we must use each digit optimally, i.e., use as many nines as needed. $s(n)$ must therefore be $\lfloor n/9 \rfloor$ nines trailing the remainder after selecting this many nines, i.e., $n - 9\lfloor n/9 \rfloor$ or $n \pmod 9$.

For example, $s(10) = 19, s(11) = 29, \ \dots \ , s(25) = 799, s(50) = 599999$. It's easy to observe

$$s(n) = (n \pmod 9 + 1) \cdot 10^{\lfloor n/9 \rfloor} - 1$$

Now, we simplify $S(k)$. Defining $\tilde{s}(n) := s(n) + 1$, we have

$$\begin{aligned}
S(k) &= \sum_{n=1}^{k} s(n) = \sum_{n=1}^{k} \tilde{s}(n) - k = \sum_{n=1}^{9\lfloor k/9 \rfloor} \tilde{s}(n) + \sum_{n=9\lfloor k/9 \rfloor+1}^{k} \tilde{s}(n) - k \\
&= \sum_{j=0}^{\lfloor k/9 \rfloor - 1} \sum_{i=0}^{8} ((i+1) \cdot 10^j) + \sum_{n=9\lfloor k/9 \rfloor+1}^{k} ((n \pmod 9) + 1) \cdot 10^{\lfloor n/9 \rfloor}) - k \\
&= \sum_{j=0}^{\lfloor k/9 \rfloor - 1} \left(10^j \sum_{i=1}^{9} i \right) + \sum_{h=1}^{k-9\lfloor k/9 \rfloor} (i+1) \cdot 10^{\lfloor k/9 \rfloor} - k \\
&= 45 \sum_{j=0}^{\lfloor k/9 \rfloor - 1} 10^j + 10^{\lfloor k/9 \rfloor} \sum_{h=2}^{k-9\lfloor k/9 \rfloor+1} i - k \\
&= 45 \frac{10^{\lfloor k/9 \rfloor} - 1}{10 - 1} + 10^{\lfloor k/9 \rfloor} \sum_{h=1}^{k \pmod 9 + 1} i - 1 - k \\
&= 5 \cdot (10^{\lfloor k/9 \rfloor} - 1) + 10^{\lfloor k/9 \rfloor} \cdot \frac{(k \pmod 9 + 1)(k \pmod 9 + 2)}{2} - 1 - k \\
&= 10^{\lfloor k/9 \rfloor} \cdot \left(5 + \frac{(r+1)(r+2)}{2} \right) - (k+6) \quad \text{for } r := k \pmod 9
\end{aligned}$$

Thus we have a closed form formula for $S(k)$. However, computing $S(k)$ directly is impractical, since for large inputs such as f_{90} (which is of the order 10^{18}), $s(k)$ has $\approx 10^{17}$ digits and $S(k)$ far more. We must therefore apply modulo m in intermediate steps of the calculation:

$$\begin{aligned}
S(k) \pmod m &\equiv \left(10^{\lfloor k/9 \rfloor} \cdot \left(5 + \frac{(r+1)(r+2)}{2} \right) - (k+6) \right) \pmod m \\
&\equiv \left(10^{\lfloor k/9 \rfloor} \pmod m \cdot \left(5 + \frac{(r+1)(r+2)}{2} \right) + (M - (k+6)) \right) \pmod m
\end{aligned}$$

where $M := m \cdot \lfloor (k+6)/m \rceil$, i.e., the smallest multiple of m larger than or equal to $k+6$.

Now, with the help of the `powermod` function, we can easily calculate $S(k) \pmod m$ for even large k. Taking the summation of $S(k)$ for the first ninety Fibonacci numbers modulo m is then trivial.

Solution

```julia
k_bound, modulo = 90, 1000000007;
```

```
generate_fibonacci (generic function with 1 method)
```

```
S (generic function with 1 method)
```

```julia
S(k, m) = (powermod(10, k ÷ 9, m) * (5 + (k % 9 + 1) * (k % 9 + 2) ÷ 2)
    + m * ((k + 6) ÷ m) - (k + 6)) % m
```

```
inverse_digit_sum (generic function with 1 method)
```

```julia
function inverse_digit_sum(k_bound, m)
    F = generate_fibonacci(k_bound)
    return sum(S.(F[2:end], m)) % m
end
```

```
922058210
```

```julia
inverse_digit_sum(k_bound, modulo)
```

Benchmark

```julia
using BenchmarkTools
```

```
BenchmarkTools.Trial: 10000 samples with 1 evaluation.
 Range (min … max):  110.800 μs …   4.073 ms  ┊ GC (min … max): 0.00% … 0.00%
 Time  (median):     135.100 μs               ┊ GC (median):    0.00%
 Time  (mean ± σ):   141.393 μs ±  50.978 μs  ┊ GC (mean ± σ):  0.00% ± 0.00%
```

```
 111 μs           Histogram: frequency by time          253 μs <

 Memory estimate: 3.20 KiB, allocs estimate: 5.
```

```julia
@benchmark inverse_digit_sum(k_bound, modulo)
```

Validation

```
brute_S (generic function with 1 method)
```

```julia
@assert S(20, modulo) == 1074
```

```julia
for i in 1:1000
    @assert S(i, modulo) == brute_S(i, modulo)
end
```

```
ladder_mod (generic function with 1 method)
```