Problem #387: Harshad Numbers

Solved 26 December, 2021

A Harshad or Niven number is a number that is divisible by the sum of its digits. 201 is a Harshad number because it is divisible by 3 (the sum of its digits.) When we truncate the last digit from 201, we get 20, which is a Harshad number. When we truncate the last digit from 20, we get 2, which is also a Harshad number. Let's call a Harshad number that, while recursively truncating the last digit, always results in a Harshad number a *right truncatable Harshad number*.

Also: 201/3=67 which is prime. Let's call a Harshad number that, when divided by the sum of its digits, results in a prime a *strong Harshad number*.

Now take the number 2011 which is prime. When we truncate the last digit from it we get 201, a strong Harshad number that is also right truncatable. Let's call such primes *strong, right truncatable Harshad primes*.

You are given that the sum of the strong, right truncatable Harshad primes less than 10000 is 90619.

Find the sum of the strong, right truncatable Harshad primes less than 10^{14}.

Solution

```
using Pkg; Pkg.activate(".")     ;
```

```
Main.workspace#2.PrimeFunctions
```

```
include("General Functions/PrimeFunctions.jl")
```

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

```
# returns true if n is a Harshad number
is_harshad(n) = n % sum(digits(n)) == 0
```

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

```
# returns all the right-truncatable Harshad (RTH) numbers less than or equal to limit
function right_truncatable_harshads(lim)
    # set up base-case: all one-digit numbers are trivially Harshad
    rth_nums = [collect(1:9)]
    # in each iteration, add RTH numbers of one larger order of magnitude
```

```
        end
        # insert new RTH numbers
        push!(rth_nums, new_rth_nums)
    end
    # concatenate and return all found RTH numbers
    return reduce(vcat, rth_nums)
end
```

```
check_prime (generic function with 3 methods)
```

```
# check whether num is a prime
function check_prime(
        num,
        primes_vec = PrimeFunctions.generatePrimes(isqrt(num)),
        primes_set = Set(primes_vec)
    )
    @assert num ≤ last(primes_vec)^2
    num ≤ last(primes_vec) && return num ∈ primes_set
    return !any(p -> num % p == 0, primes_vec)
end
```

```
strong_right_truncatable_harshads (generic function with 3 methods)
```

```
# returns all strong right-truncatable Harshad (SRTH) numbers ≤ lim
function strong_right_truncatable_harshads(lim,
        primes_vec = PrimeFunctions.generatePrimes(isqrt(lim)),
        primes_set = Set(primes_vec)
    )
    # filter RTH numbers that yield a prime when divided by sum of digits
    return filter!(
        k -> check_prime(k ÷ sum(digits(k)), primes_vec, primes_set),
        right_truncatable_harshads(lim)
    )
end
```

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

```
function srth_primes_sum(lim)
    # generate vector and set of primes necessary for primality tests up to lim
    primes_vec = PrimeFunctions.generatePrimes(isqrt(lim))
    primes_set = Set(primes_vec)
    # generate strong right-truncatable Harshad numbers
    srths = strong_right_truncatable_harshads(lim÷ 10, primes_vec, primes_set)
    @show srths
    srthp_sum = 0
    # iterate over SRTH, adding each possible digit at its end to check for primes
    for srth in srths, i in 1:min(9, lim - 10*srth - 1)
        candidate = 10srth + i
        check_prime(candidate, primes_vec, primes_set) && (srthp_sum += candidate)
```

```
using BenchmarkTools
```

```
BechmarkTools.Trial: 25 samples with 1 evaluations.
 Range (min … max):  2.096 s …    2.781 s  ┊ GC (min … max): 0.37% … 0.00%
 Time  (median):     2.396 s               ┊ GC (median):    0.10%
 Time  (mean ± σ):   2.402 s ± 188.859 ms  ┊ GC (mean ± σ):  0.11% ± 0.10%
```


```
 Memory estimate: 33.70 MiB, allocs estimate: 108530.
```

```
b = @benchmarkable srth_primes_sum(10^14); run(b, seconds=60)
```

Validation

```
@assert is_harshad(201)
```

```
@assert 201 ∈ right_truncatable_harshads(201)
```

```
@assert 201 ∈ strong_right_truncatable_harshads(201)
```

```
@assert srth_primes_sum(10000) == 90619
```