

# Limited Proximity Sieve

LPS adjusts given inputs for primality and lists nearby primes. LPS performs the sieve of Eratosthenes on an interval beginning with the given random number where modular operations with products of primes reveal values near the random number who are multiples of the product constituents, from there modular operations are continued with values of significantly reduced length. (My algorithm from 2018.) This can be used as an extremely fast probabilistic pretest in formal tests. Proof of concept:

## Fast probabilistic primality test for LPS

(proof of concept with example)

---

- Speed-up for large pq files? if  $a/pq$  then  $a/p$  and  $a/q$   
 - generate multiple primes at once:  
 test  $p$  and  $q$  with primes  $< m$   
 then test  $pq$  with primes  $< n$  where  $m < n$
- $p < 10^9$  if current version is fast.

if  $n \bmod g$  is close to  $g$ , then ~~work from~~ ~~working spot~~ the positive side of  $n$  and set " $n \bmod g$ " equal to where ~~first divides some value~~  $> n$

if  $(n \bmod g_a) < \text{largest prime in } g_a$  then... if ignorance in bulk, then speed up =  $g_a$  where  $n \bmod g_a$  is tiny!

if  $n \bmod g_a < \text{largest prime in } g_a$  then...  $P_b - (n \bmod g_a) =$   
 landing spot on LPS[], if  $P_b > (\text{LPS size} + (n \bmod g_a))$  then ignore primes in bulk.

55 p  
10000

$n = 4,601,346,927$   
 $g = 2,699,690$   
 $\frac{n}{g} = 474.38$   
 $n \bmod g = 3,693,867$

$g_a = (2 \times 3 \times 5 \times 7 \times \dots \times 19)$   
 $g_z = (911 \times \dots)$

last mod Feb 2019  
May 2019

175,000<sup>3</sup>  
 ↑  
 20,000<sup>4</sup>  
 ↑  
 5,000<sup>5</sup>  
 ↑  
 1,000<sup>6</sup>  
 ↑  
 500<sup>7</sup>  
 ↑  
 100<sup>10</sup>

$g$  is group  
 $g_a = (p_1 p_2 p_3 \dots)$   
 $p$  is unique prime

$(n - ((n \bmod g_a) \bmod p_b)) \bmod p_b = 0$   
 -or-  
 $p_b \mid n - ((n \bmod g_a) \bmod p_b)$

\* rule modification: "All primes are random  $\pm 8 \ln(n)$ " possible because SWF is performed on  $n - 200k$  where actual  $n$  is tracked

if solutions  $> 0$ , use solution near  $n$   
 if solutions  $= 0$ , use  $p$  near  $n$

↑ or retry if XF fails based on fail probability