Computing the Pseudoprimes up to 10^{13}

Michal Mikuš *

KAIVT FEI STU, Ilkovičova 3, Bratislava, Slovakia

Abstract. The paper extends the current known tables of Fermat's pseudoprimes to base 3 with the bound of 10^{13}. The paper is motivated by works of C. Pomerance (1980), G. E. Pinch (2000), William Galway (2002) and Jan Feitsma (2009) who provided tables of pseudoprimes with base 2 up to 10^{13} (Pinch), 10^{15} (Galway) and 10^{17} (Feitsma).

1 Introduction

Many public key cryptosystems (RSA, ElGamal, etc.) require a fast generation of prime numbers. The common algorithms generate odd integers and test them for primality. Because the deterministic testing algorithms are slow in practice, faster (non-deterministic) algorithms with some non-zero (but arbitrary low) error probability are used instead.

One of the fastest algorithms is the Fermat primality test; Fermat pseudoprimes being defined as such composite numbers that pass this test. More formally, we call an integer number N a Fermat pseudoprime with respect to base B, if

$$B^{N-1} \equiv 1 \pmod{N}$$

There are more sophisticated tests used in practice, such as Baille-PSW primality test, that are based on the Fermat primality test. The existing tables of pseudoprimes are used to prove the desired properties of these tests. For example the Baille-PSW test is flawless up to 10^{17} – it has been verified with the help of the pseudoprime tables that no composite integer $N < 10^{17}$ passes this test.

Our contribution and related work
The first paper in this area was due to C.Pomerance, L.Selfridge and S.Wagstaff, who computed pseudoprimes up to 25.10^9 [1]. Later R.Pinch in [2] extended their tables up to 10^{13} and provided a quite detailed documentation of the algorithms used. Further improvements were due to W.Gallway [3] and J.Feitsma [4], who computed pseudoprimes up to 10^{15} and 10^{17} respectively. To the best of our knowledge there have been no published improvements in the method since R. Pinch..

Our work was aimed at two goals: extending the pseudoprime tables with respect to base 3 and providing a complexity analysis of the algorithm described by R.Pinch. We have slightly improved this algorithm and provide results for base 3 with the bound of 10^{13}.

* This material is based upon work supported under the grant NIL-I-004 from Iceland, Lichetenstein and Norway through the EEA Financial Mechanism and the Norwegian Financial Mechanism. The computing facilities were also provided by the GRID laboratory (supported by project VG 1/0649/09) at FIIT STU.

2 Algorithm analysis

In this section we shortly describe the algorithms used for finding pseudoprimes. Fully detailed documentation is in [2]. In the second part we describe our modification that leads to slightly better time complexity of the search algorithm.

The algorithm is divided into two main phases: precomputation and search. Input of the algorithm is the bound X for pseudoprimes and base B; we implemented two cases: $X = 10^{13}$ and $B = 2$ and 3. The first one $B = 2$ for the check of our implementation and most importantly for measuring the running times of individual phases of the algorithm. The second run with base $B = 3$ was to extend the pseudoprime tables and comparation with the base 2.

2.1 Precomputation

The precomputation phase prepares tables of primes up to the bound with their multiplicative orders modulo B; the multiplicative order of p_i being the smallest number $f(p_i)$ such that $2^{f(p_i)} \equiv 1 \bmod p_i$. The precomputation is divided into three parallel parts, the longest one took us approx. 54 hours of CPU time for both bases.

2.2 Search

The search phase takes the precomputed tables of primes and their orders and output is the list of pseudoprimes up to desired bound X. The search is also divided into three distinct parts, each of them can be carried out separately:

1. squarefree pseudoprimes:
   (a) with all factors less than 10^9 (*main search*)
   (b) with one factor greater than 10^9 (*large factor*)
2. pseudoprimes with repeated factor

The technique used is an exhausting search in each case, but this categorization enables us to use some properties of pseudoprimes and dramatically reduce the search space. The running time of whole search is dominated by the *main search* – with distinct factors up to 10^9. The *main search* is further divided by the number of factors of N, which was between 2 and 10 for the bound 10^{13}. The *main search* can be parallelized for distinct d. Resulting time for each d will be shown in section 3. The rest two parts take negligible time for base 2. For detailed description of the techniques used in each case we refer the reader to [2]. We focus our attention on the modification of the search.

Modification of the search
The search for squarefree pseudoprimes is divided by a bound for large primes – R.Pinch set it to 10^9. It is clear that the smaller bound will reduce the space for *main search* while expand the space for the *large factor search*. Smaller bound

seems to be a reasonable choice because the *large factor search* takes negligible time with the bound 10^9 and the *main search* is the most time-consuming part of search. We considered lowering the bound to the 10^7.

The simple lookup in the precomputed prime tables shows that there are 423036 permitted primes up to 10^9 and 358846 permitted primes up to 10^7. Permitted primes are such p_i that $p_i \cdot f(p_i) < 10^{13}$ – the condition is a requirement for pseudoprime $N < 10^{13}$ and it reduced space for the exhaustive search significantly. By setting the bound for large primes to 10^7 we expected the time needed for main search to be reduced substantially. However, the experimental results showed less than 1% time improvement in the worst case $d = 4$. This results from the fact that the main search algorithm is much more sophisticated than the brute-force search. We have not been able to explain this result by analytical methods yet.

3 Computation results

In this section we provide the experimental results. The computations were performed on the FIIT STU GRID laboratory. On the GRID 20 AMD 64-bit processors were used, 2GHz and 1GB RAM each. All the algorithms were implemented in language C with the use of NTL library routines [6].

Main search for bounds 10^9 and 10^7
In this first part we provide experiments with the bound for large factor: 10^9 and 10^7. The time for the *large factor search* was negligible for both cases. The running time for the *main search* is shown in the following table.

d	bound 10^9 time (s)	bound 10^7 time (s)	spared (s)
2	2415	2275	140
3	19235	18968	267
4	26971	26786	185
5	15805	15737	68
6	4635	4614	21
7	783	783	0
8	102	101	1
9	11	11	0

The improvement in running time was less than expexted: for the longest run $d = 4$ we spared 0.68%. The other cases of d were computed paralelly, so the overall improvement was 0.68%.

Results for base 2
Following table displays the distribution of pseudoprimes according to the number of factors, repeated factor and large factor search for both bounds 10^9 and

10^7.

d	bound 10^9 number of PSP	bound 10^7 number of PSP
2	123621	99530
3	19807	19060
4	35232	34530
5	49470	49290
6	29063	29060
7	6306	6306
8	407	407
9	2	2
rep.f.	54	54
large.f.	277	26000

There are 264239 pseudoprimes with base 2.

Results for base 3

In the second part we show new results for base $B = 3$ along with the running time of each case of algorithm. The bound for large primes was 10^9. There are 264461 pseudoprimes up to 10^{13}.

d	number of PSP	time (s)
2	128484	4145
3	21328	20139
4	35534	22341
5	43478	12026
6	22576	3285
7	4405	402
8	242	51
9	3	4
rep.f.	7667	$\approx$ 28 hrs.
large.f.	344	≈ 0

The results are very similar to those with base $B = 2$. The running time was dominated by the repeated factor search, because of the prime 11. The integer 121 is a pseudoprime mod 3 and therefore[1] every $q \cdot 121$ is possible candidate for repeated factor pseudoprime. The situation for base 2 was easier because the smallest such prime was 1093 and the search was 10^4 times faster.

Interesting is also the observation that there is approximately the same number of pseudoprimes mod 2 and mod 3: only 222 more Fermat's pseudoprimes with base 3.

[1] For detailed explanation see [2].

4 Open questions

The obvious goal for further research is the search for pseudoprimes modulo 5, 7, etc. and comparation of their statistics. A more challenging goal is to search for larger pseudoprimes – this was done by W.Gallway and J.Feitsma. Since they haven't explained their methods, even more interesting and challenging will be improvement of the algorithm and reduction of the running time – this will also result into reaching higher bounds for pseudoprimes.

References

1. Pomerance, C., Selfridge, J.L., Wagstaff, S.S. : The pseudoprimes to 25.10^9. Math. Comp. 35, pp 1003–1026 (1980)
2. Pinch, R.G.E.: The Pseudoprimes up to 10^{13}. Lecture Notes in Computer Science 1838, pp 459–474, Springer-Verlag (1980)
3. Gallway, W.: The Pseudoprimes below 10^{15}.
   Url: `http://www.cecm.sfu.ca/Pseudoprimes/`
4. Feitsma, J.: The Pseudoprimes below 10^{17}.
   Url: `http://www.janfeitsma.nl/math/psp2/database`
5. Brillhart, J., Lehmer, D.H., Selfridge, J.L., Tuckermann, B., Wagstaff, S.S. jr: Factorizations of $b^n \pm 1$. (2nd ed) American Mathematical Society, Providence, R.I. (1988)
6. Shoup, V.: NTL: A Library for doing Number Theory.
   Url: `http://www.shoup.net/ntl/`