ON THE DISCOVERY OF THE 45TH AND 46TH KNOWN MERSENNE PRIMES

GEORGE WOLTMAN AND SCOTT KUROWSKI

Abstract. The Great Internet Mersenne Prime Search (GIMPS) has been coordinating a rigorous search for new Mersenne Prime discoveries. This paper provides a quick overview of Mersenne prime history, discusses the methods used by GIMPS to discover new primes, and presents the latest search results including two new Mersenne primes, $M_{43112609}$ and $M_{37156667}$.

1. Introduction

There is a long, rich history [11] in the search for primes of the form $M_p = 2^p - 1$, named Mersenne primes. In the early years the study of Mersenne primes led to several important advances in number theory. Since 1952, the computer has become instrumental in the search for these large primes leading to several algorithmic advances. Robinson, Riesel, Hurwitz, Gillies, Tuckerman, and Noll & Nickel, Colquitt & Welsh, Slowinski, sometimes partnering with Nelson or Gage all used the most powerful computers of their day to find new Mersenne primes. In 1996, George Woltman wrote software that allowed ordinary personal computers to search for these large prime numbers. In 1997, Woltman partnered with Scott Kurowski to coordinate these personal computers in a distributed computing architecture to accelerate the search.

For more than a dozen years, Chris Caldwell has maintained the Prime Pages web site (see [3]) with a superb and more complete history of Mersenne Primes and their discoverers.

2. Methodology

It can be trivially proven that M_p can be prime only if p is prime. There are 4 steps to efficiently determine whether a candidate M_p is prime or composite [13]. The first step is a brute force search for a small factor of M_p. The second step is to search for a factor using Pollard's P-1 factoring method. The third step is to run a Lucas-Lehmer primality test. The last step is to rerun the Lucas-Lehmer test to double-check the first test.

Brute force factoring. The brute force factoring step efficiently eliminates candidates by finding small factors. It is well-known that factors of M_p must be of the form $q = 2kp + 1$ and $q \equiv 1$ or $7 \pmod 8$ for odd p. We use these facts to form a set of potential factors for a Mersenne number. We then use a small prime sieve on these potential factors to eliminate most of the composite ones. The remaining potential factors are then tested using a left-to-right binary powering function [8]. We stop brute force factoring when the "time it takes to test a potential factor" > "the chance the potential factor divides M_p" times "the time it takes to run 2 Lucas-Lehmer tests". This is highly dependent on the machine architecture and code optimizations. For candidates we are currently testing we cease brute force factoring at 2^{68}. More than 60% of candidates are eliminated in this step.

Pollard P-1 factoring. The Pollard P-1 factoring step [10] seeks a factor q where $q-1$ is smooth. This method is well suited to Mersenne numbers because factors of M_p are $q = 2kp+1$. Thus 2 and p divide $q-1$, meaning we succeed whenever the smaller value k is smooth. We pick stage 1 and stage 2 bounds to maximize CPU time saved if a factor is found. That is, maximize "the chance of finding a factor using proposed bounds" times "cost of 2 Lucas-Lehmer primality tests" minus "cost of P-1 factoring using proposed bounds". Knuth [8] tells us how to use Dickman's function to compute the chance we will find a smooth k. This step eliminates about 5% of the remaining candidates.

Lucas-Lehmer testing. We use the Lucas-Lehmer primality test [7] and [8], which defines a sequence starting with $S_1 = 4$, and iterates $S_{n+1} = S_n^2$ - 2 (mod M_p) through the index $n = p-1$. M_p is prime if and only if $S_{p-1} = 0$ (mod M_p).

The main operation in this test is that of squaring numbers of size p bits. Schoöhage and Strassen [12] showed that a Fast Fourier Transform (FFT) can be used to square a p-bit number in O(p log p log log p) bit operations. In the early 1990s, Richard Crandall [4] found a way to halve the size of the FFT by eliminating the zero-padding that multiplication using an FFT requires, conveniently producing a result mod M_p. We implement this irrational base discrete weighted transform (IBDWT) algorithm in assembly language for maximum speed.

For best performance on today's personal computers we use floating-point FFTs rather than all-integer FFTs. It is a common misconception that all-integer FFTs are better because they are error-free. At the conclusion of each multiplication using floating point FFTs each FFT element must be rounded to the nearest integer. If the errors accumulated due to the inexact nature of floating point computations result in an element being off by more than 0.5, then we will round the element to the wrong integer. While all-integer FFTs are not subject to this roundoff error, all-integer FFTs are still be subject to the much more common memory corruption, overheating, overclocking, and other hardware errors that bedevil PCs of highly variable quality.

Double-checking. The double-checking step essentially re-runs the Lucas-Lehmer primality test. While Mersenne primes are immediately double-checked by third parties using different hardware and software, resource limitations require that Mersenne composites are retested using the same software. To double-check composites, the last 64 bits of S_{p-1} from the first test are saved and the last 64 bits of the double-checking test are also saved for comparison. A candidate is successfully double-checked when we get two matching 64 bit results. One note, rather than start the sequence with $S_1 = 4$, the client shifts S_1 by a random number of bits. The Lucas-Lehmer test is run and then S_{p-1} is adjusted to undo the initial random shift. This helps guard against programming error as the shift causes the FFT to operate on different data, making it highly unlikely that a bug in our FFT code would yield the same 64 bit result.

Distributed computing. The task of finding new Mersenne primes is an enormous undertaking. Lucas-Lehmer tests can require weeks of computer time and there are hundreds of thousands of candidates to test. The long computations and low bandwidth requirements make this ideal for distributed computing where a central server coordinates the work of thousands of clients. The server hands out a candidate for a client to test, the client processes the candidate and reports the result back to the server, and the process repeats.

Thus, in 1996, the distributed computing project Great Internet Mersenne Prime Search (GIMPS) was founded. The project was one of the pioneers [2] in creating supercomputer-scale power out of inexpensive personal computers.

Rather than have each client perform all 4 steps above, the server assigns only one of the four steps to a client. This allows the server to tailor assignments to each computer's capabilities. For example, slower clients can be assigned brute force factoring while more powerful computers can be assigned the more time-consuming Lucas-Lehmer primality tests.

We mentioned earlier that today's PCs are not entirely reliable, especially over lengthy computations. In practice, our error-rate for first-time tests is roughly 2%. This underlines the importance of the robustness double-checking provides to a distributed computing project.

Finally, while not a requirement, the central server provides "bells and whistles" to keep user interest high. This includes reports so the user can watch project progress or keep track of his own progress. A top producers report so that users can "compete" against one another for recognition. Teams allow users with common interests to pool their machines to compete against other teams.

3. Results

On August 23, 2008 the 45th known Mersenne prime, $2^{43112609} - 1$, was discovered by Edson Smith on a computer in the UCLA Mathematics Department. This prime number is 12978189 decimal digits long. It took 34 days on a 2.4 GHz Intel Core 2 Duo E6600 computer to prove this number prime. As of this writing, the prime is the largest known explicit prime of any type.

On September 6, 2008 the 46th known Mersenne prime, $2^{37156667} - 1$, was discovered by Hans-Michael Elvenich on his personal computer in Langenfeld near Cologne, Germany. This prime number is 11185272 decimal digits long. The test took 7 months running part-time on a 2.83 Ghz Intel Core 2 Duo CPU E8300.

The two primes were independently verified by Tom Duell and Rob Giltrap at Sun Microsystems using Ernst Mayer's publicly available Mlucas [9] program. They were also verified by Tony Reix of Bull SAS in Grenoble, France and Jeff Gilchrist using Guillermo Ballester Valor's publicly available Glucas [1] program.

There are now 46 known Mersenne primes, of which the last 12 were found by the GIMPS project. M_p is prime for $p = 2$, 3, 5, 7, 13, 17, 19, 31, 61, 89, 107, 127, 521, 607, 1279, 2203, 2281, 3217, 4253, 4423, 9689, 9941, 11213, 19937, 21701, 23209, 44497, 86243, 110503, 132049, 216091, 756839, 859433, 1257787, 1398269, 2976221, 3021377, 6972593, 13466917, 20996011, 24036583, 25964951, 30402457, 32582657, 37156667, 43112609.

At the time of this writing, double-checking has shown there are no undiscovered Mersenne primes below 18000000. The current status of the search is frequently updated at `http://www.mersenne.org/report_milestones`.

4. The future

The Electronic Frontier Foundation's Cooperative Computing Awards is offering $150,000 for the discovery of a prime with 100 million digits [6]. Today's fastest quad core PC takes about 2 years to test a single 100 million digit candidate. Hardware advances will be required before any serious attempts can be made. Such advances are on the way. In the last decade Intel introduced SSE for vector processing of 2 single-precision floats and SSE2 for vector processing of 2 double-precision floats. Intel has announced AVX, which promises vectors of 4, 8, and 16 double-precision floats in the future. Intel, Nvidia, and ATI have all promised graphics chips that will support even more double-precision vector processing power.

It took 9 years to advance from a 1 million digit prime discovery to a 10 million digit prime discovery. I suspect it will take at least that long before a 100 million digit prime is found. In the meantime, GIMPS will continue methodically searching for smaller new Mersenne primes.

5. Acknowledgments

While there are far too many people to list here, the authors wish to thank the thousands of people that have contributed to the GIMPS project. While most have contributed computer time, many have also contributed by writing software, recruiting others, participating in discussion forums, running mailing lists, helping new users, and hosting web pages.

References

[1] G. Ballester, *GLUCAS - Yet Another FFT*, `http://www.oxixares.com/glucas/`.
[2] J. Bohannon, *Distributed Computing: Grassroots Supercomputing*, Science, **308.5723** (6 May 2005), 810–813.
[3] C. Caldwell, *Mersenne Primes: History, Theorems and Lists*, `http://primes.utm.edu/mersenne/index.html`.
[4] R. Crandall and B. Fagin, *Discrete Weighted Transforms and Large-Integer Arithmetic*, Mathematics of Computation, 62 (1994), 305–324.
[5] R. Crandall and C. Pomerance, *Prime Numbers: A Computational Perspective*, New York: Springer-Verlag, 2000.
[6] Electronic Frontier Foundation, *Cooperative Computing Awards*, `http://www.eff.org/awards/coop`.
[7] G. H. Hardy and E. M. Wright, *An Introduction to the Theory of Numbers*, 5th Ed., Oxford, Clarendon Press, 1979.
[8] D. E. Knuth, *Seminumerical Algorithms Volume 2 of The Art of Computer Programming*, 3rd ed. Reading MA, Addison-Wesley, 1998.
[9] E. Mayer, *Mlucas - A Portable Program for Lucas-Lehmer Tests*, `http://www.mersenne.org/freesoft/MLucas.html`.
[10] P. L. Montgomery, *A Survey of Modern Integer Factorization Algorithms*, CWI Quarterly, 7 (1994), 337–366.
[11] P. Ribenboim, *The New Book of Prime Number Records*, Springer, 1996.
[12] A. Schoñhage and V. Strassen, *Schnelle Multiplikation groer Zahlen.*, Computing, 7 (1971), 208–213.
[13] G. Woltman, *GIMPS: Mathematics and Research Strategy*, `http://mersenne.org/math.htm`.

MSC2000: 11-04, 11A41

119 Magnolia St., Windermere FL, 34786
E-mail address: woltman@alum.mit.edu
URL: http://mersenne.org/

E-mail address: scott@scottkurowski.com
URL: http://scottkurowski.com