All You Wanted to Know about Irreducible and Primitive Trinomials*

Richard P. Brent
Computing Laboratory
University of Oxford

1 July 2003

*Oxford Supercomputer Centre Mini-Conference.

 OSC1t

Introduction

Irreducible and primitive polynomials over finite fields have many applications in cryptography, coding theory, random number generation etc. See, for example, the books by Golomb, Knuth (Vol. 2), and Menezes *et al.*

In this talk I will describe a new algorithm which has been used to find primitive polynomials of very high (in fact, "world record") degree over the field GF(2) of two elements.

The results can be generalised to *almost primitive* polynomials, which are (roughly speaking) polynomials with a large primitive factor.

Polynomials over GF(2)

GF(2) is just the set $\{0, 1\}$ with operations of addition and multiplication modulo 2.

Equivalently, GF(2) is the set of Boolean values $\{F, T\}$ with operations $\oplus$ (exclusive or) and & (and).

We consider polynomials over GF(2), that is, polynomials whose coefficients are in GF(2). *For the sake of brevity, we won't repeat this statement every time!*

Note that, for polynomials u, v over GF(2),

$$2u = 2v = 0 .$$

This implies that $u - v = u + v$ and

$$(u + v)^2 = u^2 + v^2 .$$

Some Definitions

We say that a polynomial $P(x)$ is *reducible* if it has nontrivial factors; otherwise it is *irreducible.*

If $P(x)$ is irreducible of degree $r > 1$, then

$$\mathrm{GF}(2^r) \approx Z_2[x]/(P(x)) ,$$

so we have a representation of the finite field $\mathrm{GF}(2^r)$ with 2^r elements. If x is generator for the multiplicative group of $Z_2[x]/(P(x))$, then we say that $P(x)$ is *primitive.*

Since the multiplicative group has order $2^r - 1$, we need to know the complete factorisation of $2^r - 1$ in order to test if an irreducible polynomial is primitive. However, if r is a *Mersenne exponent*, i.e. $2^r - 1$ is prime, then irreducibility implies primitivity.

Some Well-Known Results

The following results can be found in texts such as Lidl, Menezes et al. Here μ is the Möbius function, and ϕ is Euler's phi function.

1. $x^{2^n} + x$ is the product of all irreducible polynomials of degree $d|n$. For example,

   $$x^8 + x = x(1+x)(1+x+x^3)(1+x^2+x^3) .$$

2. Let J_n be the number of irreducible polynomials of degree n. Then

   $$\sum_{d|n} dJ_d = 2^n \text{ and } J_n = \frac{1}{n}\sum_{d|n} 2^d \mu(n/d) .$$

   In particular, if n is prime then $J_n = (2^n - 2)/n$.

3. The number of primitive polynomials of degree n is $P_n = \phi(2^n - 1)/n \le J_n$.

   In particular, if n is a Mersenne exponent, then $P_n = J_n = (2^n - 2)/n$.

The Reciprocal Polynomial

If $P(x) = \sum_{j=0}^{r} a_j x^j$ is a polynomial of degree r, with $a_0 \ne 0$, then

$$P_R(x) = x^r P(1/x) = \sum_{j=0}^{r} a_j x^{r-j}$$

is the *reciprocal polynomial*. Clearly $P(x)$ is irreducible (or primitive) iff $P_R(x)$ is irreducible (or primitive).

In particular, if

$$P(x) = 1 + x^s + x^r \ , \quad 0 < s < r$$

is a trinomial, then the reciprocal trinomial is

$$P_R(x) = 1 + x^{r-s} + x^r \ .$$

If it is convenient, we can assume that $s \le r/2$ (else consider the reciprocal trinomial).

Searching for Irreducible Polynomials

The irreducible polynomials (over GF(2), as usual) of degree r are analogous to primes with r digits. When searching for large primes we can quickly eliminate most candidates by sieving out multiples of small primes.

Similarly, when searching for irreducible polynomials, we can eliminate candidates by checking if they are divisible by irreducible polynomials of low degree. This process is called "sieving". Since it takes a small proportion of the overall computing time, we shall not describe it here.

Irreducible Trinomials

In applications we usually want irreducible or primitive polynomials with a small number of nonzero terms. Hence, we restrict attention to *trinomials* of the form

$$P(x) = P_{r,s}(x) = 1 + x^s + x^r \ , \quad 0 < s < r \ .$$

For simplicity, assume r is an odd prime.

Swan's Theorem

Swan (1962) determines the parity of the number of irreducible factors by an argument involving the discriminant (actually, Swan's theorem is a rediscovery of 19-th century results).

If r is an odd prime, then Swan's theorem implies that $P_{r,s}(x)$ has an even number of irreducible factors (and hence is reducible) if $r = \pm 3 \bmod 8$ and $s \ne 2$ or $r - 2$.

Expectation of Success

The probability that a randomly chosen polynomial of degree r is irreducible is of order $1/r$. Empirically, it seems that the same holds for trinomials of prime degree $r = \pm 1 \bmod 8$ (this condition implies that Swan's theorem is not applicable).

Thus, if we consider all s in the range $0 < s < r/2$, we expect a small constant number c of irreducible trinomials of degree r. Empirical evidence suggests that $c \approx 3.2$.

For example, considering the 523 prime $r \in [1000, 10000]$ such that $r = \pm 1 \bmod 8$, we find exactly 1683 irreducible trinomials, giving an estimate $c = 3.22 \pm 0.08$.

Searching for Irreducible Trinomials

Suppose r is an odd prime, $r = \pm 1 \bmod 8$, and sieving has failed to show that $P(x) = P_{r,s}(x)$ is reducible. The *standard algorithm* computes

$$x^{2^r} \bmod P(x)$$

by r steps of squaring and reduction, then uses the result that $P(x)$ is irreducible iff

$$x^{2^r} = x \bmod P(x) .$$

Complexity of the Standard Algorithm

Since we are working over GF(2),

$$\left(\sum_j a_j x^j\right)^2 = \sum_j a_j x^{2j} .$$

Thus, each squaring step takes $O(r)$ operations.

Each reduction step also takes $O(r)$ operations, since $P(x)$ is a trinomial and we can apply

$$x^{j+r} = x^{j+s} + x^j \bmod P(x)$$

for $j = r-2, r-3, \ldots, 0$ to reduce the result of squaring to a polynomial of degree less than r.

Thus, the complete test for reducibility of $P_{r,s}$ takes $O(r^2)$ operations, and to test all s takes $O(r^3)$ operations (assuming that sieving leaves a constant fraction of trinomials to test).

Improving the Standard Algorithm

The standard algorithm uses $2r$ bits of memory for squaring, and $2r + O(1)$ $\oplus$ operations for each reduction (we count bit-operations but in practice we perform 32 or 64 bit-operations in parallel using word-operations; this also applies to our new algorithm). Many of these operations are on bits which are necessarily zero. There is a better algorithm which avoids these redundant operations.

I don't have time to describe the better algorithm in detail today. Suffice it to say that the most time-consuming operation is interleaving two sequences of bits, analogous to interleaving two halves of a deck of cards when riffle-shuffling. If you are interested in the details, see a talk that I gave at Warwick, available from `http://www.comlab.ox.ac.uk/oucl/work/richard.brent/talks.html`.

Comparison of the Algorithms

The new algorithm has 75% fewer $\oplus$ operations than the standard algorithm.

Perhaps more significant than the number of operations is the number of memory references, which is reduced by 56%, from $8r/w + O(1)$ loads/stores to $\frac{7r}{2w} + O(1)$ loads/stores, on a machine with wordlength w bits.

Also significant on some machines is that the working set size is reduced by 25%, so memory references are more likely to be in the cache.

In practice the improvement provided by the new algorithm depends on many factors: the values of r and (to a lesser extent) s, the cache size, the compiler and compiler options used, whether inner loops are written in assembler, etc, but it is generally at least a factor of two.

Performance of the New Algorithm

Table 1 gives normalised times for the standard and new algorithms on various processors, for $r = 3021377$. The third column is the "normalised time" $c = \text{time(nsec)}/r^2$.

processor	algorithm	c
300 Mhz P-II	standard	6.31
"	new	1.64
500 Mhz P-III	"	0.77
833 Mhz P-III	"	1.66
300 Mhz SGI R12000	"	1.16
Sparc Ultra-80	"	0.89
900 Mhz Ultra-Sparc 3	"	0.54
1Ghz Alpha ES45	"	0.26

Table 1: Normalised time to test reducibility

The L2 cache size was 512KB on the P-II and P-III machines *except* only 256KB for the 833 Mhz P-III. The program was written in C, except that on PCs the inner loops were written in assembler to use the 64-bit MMX registers.

Times for Various Degrees

In Table 2 we show the time for a full reducibility test with our new algorithm and various degrees r on a machine (300 Mhz Pentium P-II) with 512KB L2 cache.

r	time T (sec)	$c = 10^9 T/r^2$
19937	0.42	1.06
44497	2.10	1.06
110503	14.4	1.18
132049	21.7	1.24
756839	812	1.42
859433	1027	1.39
3021377	15010	1.64
6972593	198000	4.10

Table 2: Time to test reducibility on a P-II

Some Primitive Trinomials

In Table 3 we give a table of primitive trinomials $x^r + x^s + 1$ where r is a Mersenne exponent (i.e. $2^r - 1$ is prime). We assume that $0 < 2s \le r$ (so $x^r + x^{r-s} + 1$ is not listed).

Results for $r < 756839$ are given by Heringa *et al.* [7]. We have confirmed these results.

The entries for $r < 3021377$ have been checked by running at least two different programs on different machines.

During this checking process, the entry with

$$r = 859433, \quad s = 170340$$

was found. This was surprising, because Kumada *et al.* [9] claimed to have searched the whole range for $r = 859433$. It turns out that Kumada *et al.* missed this entry because of a bug in their sieving routine!

Some Primitive Trinomials cont.

In Table 3, $x^r + x^s + 1$ is primitive over GF(2).

The entries for $r = 132049$ are by Heringa *et al.* Smaller primitive trinomials are listed in Heringa's paper and the references given there.

The second entry for $r = 859433$ is from Kumada *et al.*.

The other seven entries were found by Brent, Larvala and Zimmermann. The search for $r = 6972593$ is 96% complete.

r	s
132049	7000, 33912, 41469, 52549, 54454
756839	215747, 267428, 279695
859433	170340, 288477
3021377	361604, 1010202
6972593	3037958

Table 3: Primitive trinomials

Almost Primitive Trinomials

There is a large gap between some of the Mersenne exponents r for which primitive trinomials exist. For example, there are none in the interval $859433 < r < 3021377$, even though there are three Mersenne exponents in this interval. This is because Swan's theorem rules out about half of the Mersenne exponents – it rules out most exponents of the form $r = \pm 3 \bmod 8$.

The usual solution is to consider pentanomials (five nonzero terms) instead of trinomials, but a faster alternative is to use *almost primitive* trinomials.

Definition. We say that a polynomial $P(x)$ is *almost primitive* with *exponent* r and *increment* $\delta < r$ if $P(x)$ has degree $r + \delta$, $P(0) \neq 0$, and $P(x)$ has a primitive factor of degree r.

Almost Primitive Trinomials cont.

For example, the trinomial $x^{16} + x^3 + 1$ is almost primitive with exponent 13 and increment 3, because

$$x^{16} + x^3 + 1 = (x^3 + x^2 + 1)D(x),$$

where

$$D(x) = x^{13} + x^{12} + x^{11} + x^9 + x^6 + x^5 + x^4 + x^2 + 1$$

is primitive.

In Table 4 we list some almost primitive trinomials. In fact, we give at least one for each Mersenne exponent $r < 10^7$ for which no primitive trinomial of degree r exists.

For more on almost primitive trinomials and their applications, see my talk at Banff (May 2003), available at `http://www.comlab.ox.ac.uk/oucl/work/richard.brent/talks.html`.

r	δ	s	f
13	3	3	7
19	3	3	7
61	5	17	31
107	2	8, 14, 17	3
2203	3	355	7
4253	8	1806	255
		1960	85
9941	3	1077	7
11213	6	227	63
21701	3	6999, 7587	7
86243	2	2288	3
216091	12	42930	3937
1257787	3	74343	7
1398269	5	417719	21
2976221	8	1193004	85

Table 4:
Some almost primitive trinomials over GF(2).
$x^{r+\delta} + x^s + 1$ has a primitive factor of degree r;
δ is minimal; $2s \leq r + \delta$; the period $\rho = (2^r - 1)f$.

A Larger Example

We already considered the almost primitive trinomial $x^{16} + x^3 + 1$. Here we give an example with higher degree: $r = 216091$, $\delta = 12$. We have

$$x^{216103} + x^{42930} + 1 = S(x)D(x),$$

where

$$S(x) = x^{12} + x^{11} + x^5 + x^3 + 1,$$

and $D(x)$ is a (dense) primitive polynomial of degree 216091.

The factor $S(x)$ of degree 12 splits into a product of two primitive polynomials,

$$x^5 + x^4 + x^3 + x + 1 \text{ and}$$

$$x^7 + x^5 + x^4 + x^3 + x^2 + x + 1.$$

The contribution to the period from these factors is $f = \mathrm{LCM}(2^5 - 1, 2^7 - 1) = 3937$.

Use of Almost Primitive Trinomials in Random Number Generators

If $T(x) = x^{r+\delta} + x^s + 1$ is almost primitive with exponent r, we can use the corresponding linear recurrence

$$U_n = U_{n-r-\delta} + U_{n-s} \bmod 2^w$$

as a generalized Fibonacci random number generator.

The period will be a multiple of $2^r - 1$ (usually a multiple of $2^{w-1}(2^r - 1)$) provided $U_0, \ldots, U_\delta$ are odd.

The condition ensures that a recurrence with lags $\le \delta$ (corresponding to the degree-δ factor of $T(x)$) is not satisfied.

For more on random number generators, see my talk at ICCSA03 (Montreal, May 2003), available at `http://www.comlab.ox.ac.uk/oucl/work/richard.brent/talks.html`.

Acknowledgements

This is joint work with Samuli Larvala (HUT) and Paul Zimmermann (INRIA Lorraine).

Thanks are also due to Nate Begeman, Nicolas Daminelli, Shuhong Gao, Robert Hedges, Brendan McKay, Barry Mead, Mark Rodenkirch, Juan Varona, and Mike Yoder for their assistance in various ways.

Approximately 200,000 Mips-years of computer time was kindly provided by:

- APAC (Australia),
- CINES (France),
- OCCF, OUCL and OSC (Oxford).

References

[1] I. F. Blake, S. Gao and R. Lambert, Construction and distribution problems for irreducible trinomials over finite fields, preprint, July 2001.

[2] R. P. Brent, S. Larvala and P. Zimmermann, A fast algorithm for testing reducibility of trinomials mod 2 and some new primitive trinomials of degree 3021377, *Math. Comp.* **72** (2003), 1417–1441.

[3] R. P. Brent and P. Zimmermann, Random number generators with period divisible by a Mersenne prime, *Proc. ICCSA 2003*, Montreal, *LNCS* **2667** (2003), 1–10.

[4] R. P. Brent and P. Zimmermann, Algorithms for finding almost irreducible and almost primitive trinomials, *Proceedings of a Conference in Honour of Professor H. C. Williams*, Banff, Canada (May 2003), The Fields Institute, Toronto, to appear (preprint available).

[5] GIMPS, The Great Internet Mersenne Prime Search, `http://www.mersenne.org/`

[6] S. W. Golomb, *Shift register sequences,* Aegean Park Press, revised edition, 1982.

[7] J. R. Heringa, H. W. J. Blöte and A. Compagner. New primitive trinomials of Mersenne-exponent degrees for random-number generation, *International J. of Modern Physics C* **3** (1992), 561–564.

[8] D. E. Knuth, *The art of computer programming, Volume 2: Seminumerical algorithms* (third ed.), Addison-Wesley, Menlo Park, CA, 1998.

[9] T. Kumada, H. Leeb, Y. Kurita and M. Matsumoto, New primitive t-nomials ($t = 3, 5$) over GF(2) whose degree is a Mersenne exponent, *Math. Comp.* **69** (2000), 811–814.

[10] R. Lidl and H. Niederreiter, *Introduction to Finite Fields and their Applications*, Cambridge Univ. Press, Cambridge, second edition, 1994.

[11] A. J. Menezes, P. C. van Oorschot and S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, New York, 1997. `http://cacr.math.uwaterloo.ca/hac/`

[12] R. G. Swan, Factorization of polynomials over finite fields, *Pacific J. Mathematics* **12** (1962), 1099–1106.