Enumerating Powers of Primitive Roots

Subhash Kak

Department of Electrical and Computer Engineering, Louisiana State University, Baton Rouge, LA 70803

Abstract: This paper presents a method of computing successive powers of a primitive root g modulo a prime p by performing only addition. This is achieved by exploiting identities of the form $g^i = g^j \pm g^k \bmod p$. This could form a new approach to the computation of decimal sequences as well as discrete logarithms, which are important in many cryptographic applications.

Keywords: Decimal sequences, discrete logarithms

Introduction

The exponentiation of a primitive root, g, of a prime number, p, is basic to several cryptographic algorithms, as in the generation of decimal sequences that have cryptographic applications [2-5] as well as in the Diffie-Hellman key sharing protocol [8]. To compute discrete logarithms, one may develop a table connecting elements of the group to its indices with respect to some primitive element (see, e.g.[1,6,7]). The effort in the generation of a new power equals one multiplication, as in finding $g^{n+1} = g^n \times g \bmod p$. This paper presents a method of replacing each multiplication by one or just a small number of additions by considering exponentiation as a sum of suitably chosen prior calculations.

The Basic Theory

If g is an element in the group G of units of a commutative ring R with unity, and g is a root of some nontrivial polynomial $f(x) \in R[x]$ of the form

$$f(x) = x^n + \sum_{i=0}^{i=n-1} a_i x^i$$

where each $a_i \in \{-1, 0, 1\}$, then the elements of the cyclic group G can be enumerated using a linear feedback shift register (LFSR) that can be implemented using only additions and unary subtractions, i.e., negations.

To implement the LFSR it is more convenient to rewrite the polynomial relation as a recurrence relation of the form

$$x^n = \sum_{i=0}^{i=n-1} b_i x^i$$

where, again, each $b_i \in \{-1, 0, 1\}$. Now, one enumerates $g^{n-1}, \ldots, g^2, g^1, g^0$ and places $+1$ and -1 (i.e., places b^i) "taps" on those powers with nonzero coefficients so that only $t-1$ additions are necessary to compute g^{s+1} once g^s has been computed, where t is the number of nonzero coefficients in the recurrence relation.

For any prime there exist many identities of the kind

$$g^i = g^j + g^k \bmod p \qquad (1)$$

By multiplying both sides by g^{p-i-1}, and noting that $g^{p-1} \bmod p = 1$, we obtain:

$$1 = g^{p-i+j-1} + g^{p-i+k-1} \bmod p \qquad (2)$$

Therefore, any two values that add up to one, or differ by one (if the addition on the right hand side is replaced by subtraction), can be used to find i and j. This lends itself to simple shift register implementation.

Shift Register Implementations

In general, for any p, if $x^{p-1} - 1$ has a trinomial factor, that can be used for the implementation of the shift register in the most simplest arrangement.

Let g be a primitive element in the ring $\mathbb{Z}_p$ where p is a prime. We will now consider three examples:

Example1. Consider, p=19 and g= 13. A search of the first four powers of 13, that is 1, 13, 17, and 12, implying the degree 3 trinomial $x^3 - x + 1$, which gives us the following relation:

$$13^3 = 13^1 - 13^0 \bmod 19 \qquad (3)$$

which means that $x^3 = 0x^2 + (1)\,x^1 + (-1)\,x^0$. This leads to the implementation of Figure 1.

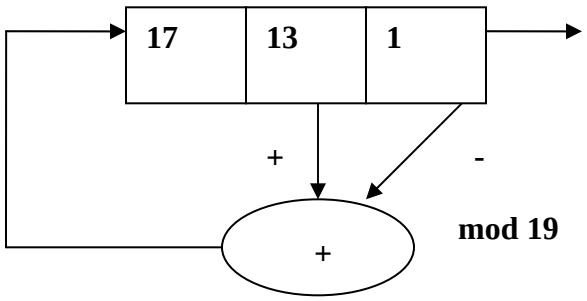


Figure 1. An implementation of successive powers of 13 mod 19 using a single addition.

It is easy to check that it will generate the next numbers 12, 4, 14, 11, 10, 16, 18, 6, 2, 7, 15, 5, 8, 9, 3 correctly.

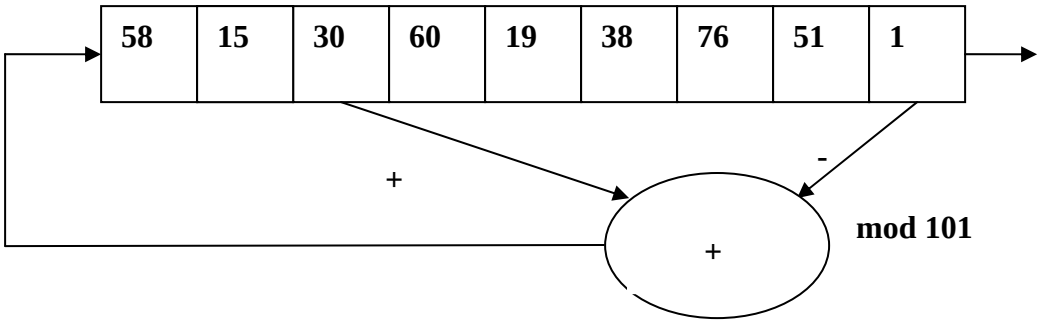
For $p = 19$, there are $\phi(18) = 6$ primitive elements, namely, *2, 3, 10, 13, 14 and 15.* All these primitives are roots of the degree *6* trinomial $x^6 - x^3 + 1$ which is a factor of $x^{18}-1$. Note here that $x^{18} - 1 = (x^9 - 1)(x^3 + 1)(x^6 - x^3 + 1)$ and *g,* being primitive, can be a root of the last factor only.

Trinomials of degree two exist for this example. For instance, $g = 14$ is a root of $g^2 + g - 1$ and $g = 15$ is a root of $g^2 - g - 1$.

Example 2. Consider g=51 and p=101. A quick calculation shows that

$$51^9 = 51^6 - 50^0 \bmod 101$$

This means that $x^9 = x^6 + (-1)\, x^0$ may be implemented by the shift register of Figure 2.



.

Figure 2. Implementation of $51^i \bmod 101$

Simple additions mod 101 will show that the sequence 1, 51, 76, 38, 19, 60, 30, 58 as loaded into the shift register will produce the next values as

29, 65, 83, 92, 46, 23, 62, 31, and so on.

Example 3. We consider now an implementation with three feedback links. Let g be so chosen that $g^2 = (g+1)/(g-1)$. This is equivalent to $g^3 = g^2 + g + 1$. Equivalently, we can write that

$$g^{n+1} = g^n + g^{n-1} + g^{n-2}$$

As example, consider $p=19$. A primitive root that satisfies the condition $g^2 = (g+1)/(g-1)$ is 13. The powers of 13 mod 19 are:

1 13 17 12 4 14 11 10 16 18 6 2 7 15 5 8 9 3 1

It is easy to confirm that

$1+13+17 \bmod 19 = 12$
$13+17+12 \bmod 19 = 4$ and so on.

Therefore, each new value in the series can be obtained by a simple addition of three terms. This may be visualized in the shift register implementation of Figure 3.

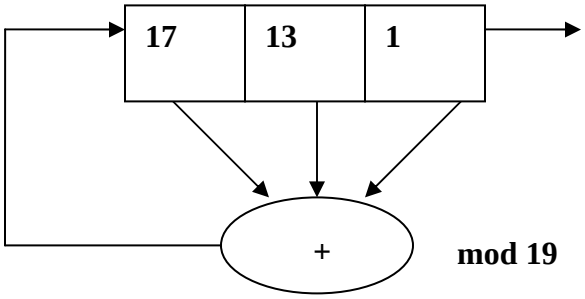


Figure 3. Implementation of $13^i \bmod 19$.

Similar conditions may be defined for sums of more than three terms.

Conclusions

Converting a multiplication into an addition can provide computational savings that can be helpful in many cryptographic calculations. In particular, it will speed up the computation of exponentials, which are required, for example, in the generation of a decimal sequence [2-6]. It may also provide help in the solution of some discrete logarithm problems.

References

1. M. Hellman and J. Reyneri, Fast computation of discrete logarithms in *GF*(*q*). In *Advances in Cryptology: Proceedings of Crypto '82*. Plenum Press, 1983.

2. S. Kak and A. Chatterjee, On Decimal Sequences. *IEEE Transactions on Information Theory*, IT-27: 647 – 652, 1981.

3. S. Kak, Encryption and error-correction coding using D sequences. *IEEE Transactions on Computers*, C-34: 803-809, 1985.

4. S. Kak, New results on d-sequences. *Electronics Letters*, 23: 617, 1987.

5. N. Mandhani and S. Kak, Watermarking using decimal sequences. *Cryptologia*, vol. 24: 50-58, 2005.

6. A.M. Odlyzko, Discrete logarithms: The past and the future**.** *Designs, Codes, and Cryptography*, 19: 129-145, 2000..
http://www.dtc.umn.edu/~odlyzko/doc/discrete.logs.future.pdf

7. S.S. Pohlig and M.Hellman, An improved algorithm for computing logarithms over GF(p) and its cryptographic significance. *IEEE Trans. on Information Theory, IT-24*, 106-110, 1978.

8. W. Stallings, *Cryptography and Network Security*. Pearson Education, Upper Saddle River, N.J., 2003.

January 10, 2005