Finding Primitive Elements in $\mathbb{Z}_p$

D. R. Stinson

In many cryptographic protocols, it is necessary to find a primitive element $\alpha \in \mathbb{Z}_p$, where p is prime. This is not too difficult to do if the factorization of $p-1$ is known. For the remainder of this note, let us assume that the factorization of $p-1$ is

$$p - 1 = {p_1}^{e_1} \dots {p_k}^{e_k},$$

where $p_1, \dots, p_k$ are distinct primes.

First, we present a lemma that provides a method of determining if a given element $\alpha \in {\mathbb{Z}_p}^*$ is primitive.

Lemma 1 *Suppose p is prime, and the factorization of $p-1$ is as given above. Then $\alpha \in {\mathbb{Z}_p}^*$ is a primitive element if and only if*

$$\alpha^{(p-1)/p_j} \not\equiv 1 \bmod p$$

for $1 \leq j \leq k$.

Proof. Let d denote the order of α. We know that d is a divisor of $p-1$ and α is primitive if and only if $d = p-1$.

First, suppose that $\alpha^{(p-1)/p_j} \equiv 1 \bmod p$ for some j. Then clearly $d \leq (p-1)/p_j$, so certainly $d \neq p-1$.

Conversely, suppose that $\alpha^{(p-1)/p_j} \not\equiv 1 \bmod p$ for $1 \leq j \leq k$. Suppose that $d \neq p-1$. Since d is a divisor of $p-1$ and $d < p-1$, there exists a prime p_j $(1 \leq j \leq k)$ such that p_j is a divisor of $(p-1)/d$. But this implies that d is a divisor of $(p-1)/p_j$. Hence, it follows that

$$\alpha^{(p-1)/p_j} \equiv \alpha^d \equiv 1 \bmod p,$$

which is a contradiction. This proves that $d = p-1$, as desired. $\square$

Now, given that we have an efficient method of determining if a given element α is primitive, how do we go about finding a primitive element? This can be done quite easily by means of a Las Vegas algorithm, by choosing random values for α and testing them, until a primitive element is found. The effectiveness of this approach depends on the probability that a random element $\alpha \in {\mathbb{Z}_p}^*$ is primitive. We observed on page 123 of the text that there are exactly $\phi(p-1)$ primitive elements in ${\mathbb{Z}_p}^*$, so the probability that a random element α is a primitive element is $\phi(p-1)/(p-1)$.

One special case of interest is when $p = 2p_1 + 1$, where p_1 is prime. In this case, the following corollary is obtained.

Corollary 2 *Suppose p and p_1 are prime, and $p = 2p_1+1$. Suppose $\alpha \in {\mathbb{Z}_p}^*$ and $\alpha \not\equiv \pm 1 \bmod p$. Then α is a primitive element if and only if $\alpha^{(p-1)/2} \not\equiv 1 \bmod p$.*

Proof. Observe that $\alpha^{(p-1)/p_1} \equiv \alpha^2 \bmod p$, and $\alpha^2 \equiv 1 \bmod p$ if and only if $\alpha \equiv \pm 1 \bmod p$. Hence the result follows from Lemma 1. ▯

We can in fact say a bit more in this case. If $\alpha \not\equiv \pm 1 \bmod p$ and α is not primitive, then $\alpha^{(p-1)/2} \equiv 1 \bmod p$. But then we have

$$\begin{aligned}
(-\alpha)^{(p-1)/2} &\equiv (-1)^{(p-1)/2}\alpha^{(p-1)/2} \bmod p \\
&\equiv (-1)^{(p-1)/2} \bmod p \\
&\equiv -1 \bmod p.
\end{aligned}$$

Thus, by Corollary 2 $-\alpha$ must be primitive.

We record this result as follows:

Corollary 3 *Suppose p and p_1 are prime, and $p = 2p_1+1$. Suppose $\alpha \in {\mathbb{Z}_p}^*$ is not a primitive element, and $\alpha \not\equiv \pm 1 \bmod p$. Then $-\alpha$ is a primitive element.*

This means that we have an efficient deterministic algorithm to find a primitive element for when p and $(p-1)/2$ are both prime.

It is not so easy to verify that elements are primitive if the factorization of $p-1$ is not known. For this reason, the designer of a cryptosystem will often construct p in such a way that the factorization of $p-1$ is known. For example, it is often desirable to implement a cryptosystem in $\mathbb{Z}_p$, where $p = 2p_1 + 1$ and p and p_1 are both prime (this is the special case discussed above). One reason why this might be done is that it ensures that the system will not be vulnerable to a Pohlig-Hellman attack on the discrete logarithm problem. To find such a p, the designer of the system will choose a random odd value p_1, and test both p_1 and $p = 2p_1 + 1$ for primiality using one of the probabilistic primality tests described in Chapter 4 of the text. If either of p or p_1 is found to be composite, then a new random value for p_1 is chosen and the process is repeated.

As another example, several protocols are implemented in $\mathbb{Z}_p$ where $p-1$ has a prime divisor q of a specified size (the digital signature standard is an example of this). A convenient realization of such a system would be to take $p = 2qp_1 + 1$, where p, q and p_1 are all primes. If q is to be a 160-bit prime and p is to be a 512-bit prime (as in the DSS), then p_1 will be a prime of approximately 352 bits. Here, the designer of the system would begin by choosing random values q and p_1 of the appropriate size, and then define $p = 2qp_1 + 1$. The three integers p, q and p_1 will all be tested for primality using a probabilistic proimality testing algorithm.