5. Factorisation and Certificates

After running the Miller-Rabin test we are either armed with the "certainty" (as far as internal error-checking mechanisms within the computer permit us to be certain!) that some number N is composite or a very high probability that it is indeed prime. For some situations neither of these is enough. We wish to offer an explicit "proof" by way of a factorisation of the number in the composite case; or by a certificate or "witness" to its primality in the second case. Such certificates may also be required if we wish to prove that the factorisation we have obtained is complete!

5.1. **Pollard's ρ.** We saw that the trial division technique was undermined by the requirement of a large number of primes *and* the number of trials that need to be performed. While this made it slow for "testing" primality or compositeness, we have not offered any alternative to it (so far) for the purpose of finding prime factors. The method now presented is quite a practical alternative; this speed has some theoretical basis as well; unfortunately, that theoretical basis is incomplete, so the algorithm *may* be slower than trial division in some cases.

The idea behind this method is that iterated self-maps of finite sets must cycle. Let S be a finite set, $f : S \to S$ be *any* map and $x_0 \in S$ be some starting point. We define $x_{k+1} = f(x_k)$ for $k \geq 0$. By the finiteness of S there is some pair (p, q) so that $x_{p+q} = x_p$; but then by applying f repeatedly to both sides it is clear that $x_{r+q} = x_r$ for all $r \geq p$. The smallest p so that x_p is repeated is called the pre-period M; the smallest q is called the period T (these depend to some extent on x_0 as well as f and S). The points $x_0, \ldots, x_M$ are the "tail" and the points x_{M+r}, $r \geq 0$ are the "head" the Pollard's ρ (the name is given for the shape of the letter). Clearly, determining M and T (given S, f and x_0) is an interesting computational problem. Before that let us see what this has to do with factoring.

Now suppose $S = S_1 \times S_2$ and $f = (f_1, f_2)$, then f_1 (respectively) f_2) will have its own (M_1, T_1) (respectively (M_2, T_2)) as pre-period and period. Each will (in general) be less than that for S; certainly those for S are upper bounds.

In particular, let us look at the case where $S = \mathbb{Z}/N\mathbb{Z}$, when N is composite; we know that there are $S_1 = \mathbb{Z}/a\mathbb{Z}$ and $S_2 = \mathbb{Z}/b\mathbb{Z}$, where $N = ab$ with a and b co-prime. Thus we should look for T_1 (or T_2). We know we will have found a period when $\gcd(x_{p+q} - x_p, N) > 1$. If this GCD is less than N then we have found a factor (and T_1 is a multiple of q) otherwise we have only found a multiple of T; hopefully we will not be so unlucky!

Another aspect to examine is what kind of maps f are "good" from the point of view of finding M and T. Clearly, we can divide S into the set of repeating points S_{cyclic} and the set of transients $S_{\text{transient}}$ (which are never repeated). If the latter set is very large, then M is likely to be very large. On the other hand if the former set is very large it is also likely that T is large. Heuristic analysis asserts that for a "randomly chosen map" f (i. e. a "random" element of $\text{Hom}(S, S)$) M and T are bounded by the condition $M + T \leq \sqrt{\#(S)}$ for $\#(S)$ large.

Randomly chosen maps may not be good for us since we need the map to have the form (f_1, f_2). In the case when $S = \mathbb{Z}/N\mathbb{Z}$ this condition $f = (f_1, f_2)$ can be easily ensured by taking f to be a polynomial map (Chinese Remainder Theorem once again!). However, *every* map on S is a polynomial map when N is prime so we can expect that polynomial maps are adequate for our purposes.

Now let us see how we can determine M and T (actually we are looking for T_1 but that aspect has been explained earlier so we will ignore it here). Clearly, storing all the iterates x_k and comparing them until we find a match is impractical when M and T are large.

The original method suggested by Pollard and Floyd was as follows. Let us compute also the sequence $y_0 = x_0$ and $y_{k+1} = f(f(y_k))$. It is clear, by induction that $y_k = x_{2k}$. Thus, if k is a multiple of T, we will get $y_k = x_k$. By checking for this identity at each iterative step we can find a multiple of T. Of course, because of the transient M it is unlikely that we will find T, but if M and T are of comparable size then we will find a small multiple of T this way. Another problem with this approach is that we need to compute the function f three times at each iterate and that can sometimes be considered expensive.

Another approach was suggested by Brent. Let us first try to look for the "size" of M and T. Thus, if M and T have n bits, then we should find a repetition for $k = 2^n - 1$ and $k + T = 2^n + T - 1$, the latter being less than $2^{n+1} - 1$. Thus, we store $y_n = x_{2^n-1}$ and compare it with x_k when k lies between 2^n and $2^{n+1} - 2$. It is clear how we can iterate over this procedure. This procedure has only one computation of f for each iteration. On the other hand, we are over-estimating M by a (worst-case) factor of about 2, which means we are making twice as many tests as in the Pollard-Floyd method. Clearly, the choice between the two methods depends on which is more time consuming—function computation or comparison.

A further improvement to the Brent method is possible if we note that when we are checking for repetitions between $k = 2^n - 1$ and some k between 2^n and $2^{n+1} - 2$, we have *already* checked for periods of size $n - 1$ bits (ignoring M for the moment). Thus we can start making comparisons only after we cross the half-way mark $2^n + 2^{n-1} - 1$. Because of M (transients again!) we may actually *not* have checked the periods and so we will only obtain multiples of T if we do this; but we will have saved exactly half the comparisons in return!

This observation also fits in well with Pollard's idea of reducing the number of comparisons in his factorisation method as follows. Instead of computing $\gcd(x_k - y_n, N)$ at each iteration, he takes the product of $x_k - y_n$ over (say) 10 iterations of k and computing GCD only in time in 10. This reduces the number of comparisons as well.

To make an algorithm we must choose algebraic self-maps f on $\mathbb{Z}/N\mathbb{Z}$. It is clear, that linear maps will have periods equal to the size of the prime factors so we may as well have used trial division. We take the next thing that comes to hand which is a map like $f(x) = x^2 + 1$ and hope it is a "random enough" choice! Powering maps like $x \mapsto x^2$, are better studied via the $(p - 1)$ method which we will see later—in particular, the periodicity of these maps has to do with a factorisation of $(p - 1)$ when p is a factor of N. Thus, we will stick with quadratic maps and hope that this is good enough[2]; this is similar to the choice of "small" numbers as a base in the Miller-Rabin test with one crucial difference—the outcome of the Pollard ρ will be a "real" factorisation, not a probabilistic one. Finally, if we are unsuccessful (in finding a factorisation) with a given f and x_0 we need to vary *both* and not just x_0.

After all that verbiage (which is used to disguise the fact that we are not really sure of the justifications!), let us come to the algorithm. Pick a small constant

[2]However, note that $x \mapsto x^2 - 2$ is in fact a powering map in disguise. If we put $x = y + 1/y$ then this is the same as $y \mapsto y^2$

c other than 0 and 2 and consider the function $f(x) = x^2 - c$. Pick a point x_0 in $\mathbb{Z}/N\mathbb{Z}$ (usually one of small size). Pick a small number s of steps (usually $s = 20$). Let $y_0 = 0$, $e = 0$ and $P = 1$ (e will count the number of bits in M and T). While k is between 2^e and $2^e + 2^{e-1} - 1$, we set $x_{k+1} = f(x_k)$. While k is between $2^e + 2^{e-1}$ and $2^{e+1} - 2$, we set $x_{k+1} = f(x_k)$ *and* multiply P by $(x_k - y_e)$. Every s steps we compute $\gcd(P, N)$. If this is greater than 1, then we have found a period (somewhere in the last s steps); otherwise we set P to be 1 again and continue. If we found a GCD, we set $z_0 = y_e$ and iteratively compute (at most s times) $z_{l+1} = f(z_l)$ and $\gcd(x_k - z_l, N)$. We will find a non-trivial GCD for some l between 0 and $s - 1$. If this GCD is less than N then we have found a factor; else we have been unsuccessful, so we change c and x_0. If we found a factor a then we can continue, replacing N with N/a, starting with the given tuple (e, k, x_k, y_e); we need not start at the beginning since periods smaller than the one found have already been (essentially) excluded. Note that all arithmetic operations (except GCD and subscript arithmetic!) are to be done modulo N/a in this continuation.

5.2. **Group theoretic method.** In the Miller-Rabin test we used the group of units in $\mathbb{Z}/N\mathbb{Z}$ (and we will continue to do so) but in fact any algebraic group scheme G can be used to study the factorisation or primality of N. Such a scheme assigns a group $G(N)$ to an integer N. By an application of the Chinese Remainder theorem (to $\mathbb{Z}/N\mathbb{Z}$ and not to the group!) one can show that $G(N)$ is a product of the groups $G(a)$ and $G(b)$ if $N = ab$ with a and b coprime. Another aspect is that Hensel's lemma allows us to write (for all but a finite set of "small" primes p) $G(p^e)$ as a product of $G(p)$ and a group *naturally* isomorphic to a direct sum of $\mathbb{Z}/p^{e-1}\mathbb{Z}$'s. The group theoretic method for factorisation depends on the possibility that $G(p)$ has a particularly simple structure for some prime p that divides N. One such criterion for simplicity is *smoothness*.

Definition 1. Let B be an integer. An integer N is said to be *B-powersmooth* if it is a product of coprime numbers less than B. Equivalently, N of product of prime powers, each of which is less than B.

Definition 2. Let B be an integer. An integer N is said to be *B-smooth* it is a product of primes less than B.

We note that a number N is B-powersmooth if and only if it divides the least common multiple $l_B = \mathrm{lcm}(1, \ldots, B)$ of all numbers less than B, whereas a B-smooth number can be arbitrarily large when $B > 1$ (for example 2^n is B-smooth in this case).

The claim is that there are quick procedures to factor N if it has a prime factor so that $G(p)$ is B-powersmooth or more generally has a "large" B-smooth number as factor. We now restrict ourselves to the case when G is the group of units to understand this procedure.

Let us now assume that N has a prime factor p so that $p-1$ (which is the order of the group of units in $\mathbb{Z}/p\mathbb{Z}$) is B-powersmooth. We want to find this prime factor. The technique is to take a "random" x in $\mathbb{Z}/N\mathbb{Z}$ and calculate $\gcd(x^a - 1, N)$ for a dividing B. Whenever this is not 1 or N we have hit on a factorisation of N. As usual, we use Pollard's idea of accumulating numbers to avoid computing GCD too often.

Let $L = (l_1, \ldots, l_k)$ be a list of all primes less than or equal to B. We pick an x in $\mathbb{Z}/N\mathbb{Z}$ (which is usual taken to be "small"). We also a pick an s which is the

number of steps over which we will accumulate. We initialise our accumulator P as 1. We also initialise by setting $i = 1$ so that we pick the first prime. We now loop over the following steps.

We first keep a y and j so that we can backtrack over these s steps; these are initialised as x and i. We compute the largest power p_i of l_i that is less than B and replace x by x^{p_i}. We multiple P by $x - 1$. We then increment i. Every s steps (or if i becomes k), we compute $\gcd(P, N)$. If this is 1, then we re-initialise y and j to be the current values of x and i respectively and set $P = 1$ and loop (unless i is k in which case we have *proved* that N is *not* divisible by a p so that $p - 1$ is a B-powersmooth number!). Otherwise, we have found a non-trivial GCD, for some power of y which divides the powers $p_j, \ldots, p_i$ (here i is at most $j + s$).

Now, we set $P = 1$, and starting with $m = j$ do the following. Replace y by its p_m-th power y and check $\gcd(y - 1, N)$. We increment m and continue. We know that for some $k \leq i$, we will find a non-trivial GCD. If this is N, then we know that N *does* have a factor p so that $p - 1$ is B-powersmooth so we should try again with some other choice of x. What has happened in this case is that the order of the chosen x has coincided in the group of units modulo different factors of N; so a different choice of x should do the trick.

Even when the above computation *proves* that N has no B-powersmooth factors, the above computation should not be thrown away! There is a second stage process which examines the case when N has a factor p so that $p - 1$ is the product of a B-powersmooth number and a prime number less than B^2 (in other words $p - 1$ is completely factored by trial division upto B). More specifically, let us assume the $p - 1$ is of the form fq where q is a prime less than C and f is B-powersmooth (where $C >> B$). We keep the list $D = (d_{k+1}, \ldots, d_l)$ of successive differences for primes between B and C (i. e. $d_{k+1} = l_{k+1} - l_k$ and so on). One we have exited from the previous algorithm, we put $b = x$ and compute the list of powers b^{d_j}. We replace x by x^{l_k} and set $i = k$ and then loop as follows.

We set our accumulator P to 1, y to x and j to i (which are for backtracking as before). We increment i and multiple x by b^{d_i}, (which we have already computed). Then we multiple P by $(x - 1)$. Every s steps (or if i becomes l), we compute $\gcd(P, N)$. It this GCD is 1, we loop back (or if $i = l$ then we have shown that N does not have a prime facor of the required form). Otherwise, there is some prime between p_j and p_i which is like q above.

We continue the analysis at this level as follows. We start at $m = j$ and multiply y by b^{d_m} and check $\gcd(y - 1, N)$. If this is 1, then we increment m and continue (we will do this at most s times). Otherwise we have found a non-trivial GCD. If this is not N, then we have a factor. On the other hand, if this is N, then as before we must take a different starting x at stage 1 and repeat the process, because we have shown that there *is* a factor p of the reuqired form.

We can replace the B-powersmooth-ness condition above by B-smoothness since we have an upper bound $\sqrt{N}$ for the powers in any case. Thus an appropriate modification to stage 1 (call it stage $1\frac{1}{2}$!) will alow us to incorporate B-smooth factorisations as well. This approach will replace a constant (essentially $\log(B)$) in the complexity (number of steps in terms of $\log(N)$) by $\log(N)$. Thus the order of complexity is increased by 1.

5.3. **Primality Certificates.** We now examine the situation where N is almost certainly prime (having passed the Miller-Rabin test many times with flying colors!).

In such a situation, we wish to provide a "certificate" that N is a prime. In unison with Knuth, we could ask "Why?". After all, it is really so highly improbable that N is not a prime that this is not worth considering. One situation that one can think of is where an "oracle" produces keys for us. While we trust the oracle not to "leak" a key, we are not so sure that the oracle (in order to save time and money) may be using some quick and dirty method to generate the modulus, which may be weak. Then we would ask the oracle to "provide proof" that it has given us a prime number. Another situation is that someone "pays" us to factor a number—we would need to certify that the factorisation is complete. The certificate should be very "easy" to check.

From this point of view, it is no use saying "it passed the Miller-Rabin test for me why don't you try it". In fact (somewhat more surprisingly perhaps) it is no use saying "I have tried all divisors upto $\sqrt{N}$". Thus even if you are "convinced" that you have a prime; how can you convince someone else without asking the other person to perform an identical computation!

Again group theoretic methods are very useful. In the situation of the group scheme as above, suppose we find an non-trivial element g in $G(N)$ whose order m is larger than the order of $G(p)$ for any prime less than $\sqrt{N}$ (recall that we have a good estimate of the order of $G(p)$). Let $d < m$, then there is some "co-ordinate" of g^d which differs from the same coordinate for the identity element of G; this is what one means by saying that g^d is not the identity element of $G(N)$. Thus, this difference must be a unit of $\mathbb{Z}/N\mathbb{Z}$ (since we are morally certain that N is a prime!). In particular, for every prime factor q of m we provide a co-ordinate of $g^{m/q}$ and its inverse x_q in $\mathbb{Z}/N\mathbb{Z}$. This is a certificate of primality.

The person receiving this certificate would argue as follows. Suppose p is the a prime factor of N that it is less than $\sqrt{N}$. Then some power g^d for $d = m/q$ must become trivial in $G(p)$ (since a this group cannot have an element of order large than it!). But the given co-ordinate of g^d differs from the same co-ordinate of identity by a unit in $\mathbb{Z}/N\mathbb{Z}$ (we have given a proof of this by providing x_q). Thus this co-ordinate cannot be zero. Hence there are not such prime prime factors—hence N is prime.

Thus a primality "certificate" would be the tuple $(G, g, m, \{x_q\}_{q|})$. Of course, the correct-ness of this certificate depends on the primality of various q's, so we would need a certificate for those as well!

To make this explicit, suppose we find an element a in $\mathbb{Z}/N\mathbb{Z}$ so that $a^m = 1$ but $a^{m/q} \neq 1$ and $q \geq \sqrt{N}$; for some integer m and a prime factor q. Then, N is prime; for if p is a factor of N, then the $\gcd(a^{m/q} - 1, p)$ divides $\gcd(a^{m/q} - 1, N)$ and thus $a^{m/q} \neq 1$ in $\mathbb{Z}/p\mathbb{Z}$. But that means q divides $p - 1$. On the other hand, if N is not prime, N msut have a prime factor smaller than $\sqrt{N}$.

This can be carried some steps further.

Proposition 12. *Suppose that $N - 1 = f \cdot r$, where f is completely factored into primes p_i, r has all its factors greater than B and $\gcd(f, r) = 1$. Moreover, suppose that $f \cdot B \geq \sqrt{N}$. Now suppose that we find a_i so that its order is a multiple of the exact power of p_i that divides $N - 1$. Moreover, we have b so that $b^{N-1} = 1$ and $\gcd(b^f - 1, N) = 1$, then N is a prime. Conversely, given such a factorisation of $N - 1$, there are a's as required.*

Proof. Let d be a prime divisor of N. Then, the order of a_i in $\mathbb{Z}/d\mathbb{Z}$ is divisible by exactly the same power of p_i that divides $N-1$ ($\gcd(a^{p_i^{e_i-1}}-1, N) = 1$ implies the same with N replaced by d). This means that f divides $d-1$. Now, let e be the order of a_0 in the units of $\mathbb{Z}/d\mathbb{Z}$. Then e divides $d-1$ and $N-1 = f \cdot r$ and does not divide $f = (N-1)/r$. Thus $\gcd(e, r) > 1$. In particular, $\gcd(e, u) > B$ (since every prime factor of r has this property). Thus $\gcd(e, u) \cdot f$ divides $d-1$, so that d becomes larger than $\sqrt{N}$. But this cannot be true for every prime factor of N unless N is prime. (Exercise: find a more group-theoretic proof). □

This can be used to provide a primality certificate as follows. We use trial division upto a bound B_0 to obtain a factorisation $N-1 = fr$. If $fB_0 \geq \sqrt{N}$, then we take $B = B_0$ and we are done. Otherwise we check for the primality of r (say using the Miller-Rabin test). If it is prime then we are done again (we have a complete factorisation of $F = N-1$). Otherwise we increase the bound B and continue. The main problem (as usual) is with $N-1$ having a few large prime factors; in this case we would have to proceed to a bound B like $\sqrt{N}$. As it turns out, once we have a factorisation of $N-1$, then we can proceed more surely, testing (in succession $a^{N-1/q}$ for each prime factor q of $N-1$; if we fail for a given a we can continue with the next a). One can show that the latter will succeed quite quickly.

Lemma 13. *Let q be a prime so that q^f exactly divides the order of a cyclic group G. The collection of elements, whose order is exactly divisible by q^e for $e < f$ has cardinality at most $1/q$ times the cardinality of G.*

Thus the probability that a given element a will fail for all q is the product of $(1-1/q)$ which is very small. The proof of the lemma is easy and left as an exercise.

As we can see the main stumbling block for this method is that we need to factor $N-1$ since the units in $\mathbb{Z}/N\mathbb{Z}$ is the *only* group available with us (so far) to apply this method to. Later we will expand the class of groups (and so we can try to *pick* a group with order easily factorisable) and it will be easier to find such primality certificates.