of the time is congruent to -1 rather than $+1$ modulo q.

(ii) $m/2$ is not a multiple of either $p-1$ or $q-1$. In this case $a^{m/2}$ is $\equiv 1$ modulo both p and q (and hence modulo n) exactly 25% of the time, it is $\equiv -1$ modulo both p and q exactly 25% of the time, and for the remaining 50% of the values of a it is $\equiv 1$ modulo one of the primes and $\equiv -1$ modulo the other prime.

Thus, by trying a's at random with high probability we will soon find an a for which $a^{m/2}-1$ is divisible by one of the two primes (say, p) but not the other. (Each randomly selected a has a 50% chance of satisfying this statement.) Once we find such an a we can immediately factor n, because $g.c.d.(n,\ a^{m/2}-1)=p$.

The above procedure is an example of a *probabilistic algorithm*. We shall encounter other probabilistic algorithms in the next chapter.

3. How do we send a signature in RSA? When discussing authentication in the last section, we assumed for simplicity that $\mathcal{P}=\mathcal{C}$. We have a slightly more complicated set-up in RSA. Here is one way to avoid the problem of different n_A's and different block sizes (k, the number of letters in a plaintext message unit, being less than ℓ, the number of letters in a ciphertext message unit). Suppose that, as in the last section, Alice is sending her signature (some plaintext P) to Bob. She knows Bob's enciphering key $K_{E,B}=(n_B,e_B)$ and her own deciphering key $K_{D,A}=(n_A,d_A)$. What she does is send $f_Bf_A^{-1}(P)$ if $n_A<n_B$, or else $f_A^{-1}f_B(P)$ if $n_A>n_B$. That is, in the former case she takes the least positive residue of P^{d_A} modulo n_A; then, regarding that number modulo n_B, she computes $(P^{d_A}\ mod\ n_A)^{e_B}\ mod\ n_B$, which she sends as a ciphertext message unit. In the case $n_A>n_B$, she first computes $P^{e_B}\ mod\ n_B$ and then, working modulo n_A, she raises this to the d_A-th power. Clearly, Bob can verify the authenticity of the message in the first case by raising to the d_B-th power modulo n_B and then to the e_A-th power modulo n_A; in the second case he does these two operations in the reverse order.

Exercises

1. Suppose that the following 40-letter alphabet is used for all plaintexts and ciphertexts: A—Z with numerical equivalents 0—25, blank=26, .=27, ?=28, \$=29, the numerals 0—9 with numerical equivalents 30—39. Suppose that plaintext message units are digraphs and ciphertext message units are trigraphs (i.e., $k=2$, $\ell=3$, $40^2<n_A<40^3$ for all n_A).

   (a) Send the message "SEND \$7500" to a user whose enciphering key is $(n_A,e_A)=(2047,179)$.

   (b) Break the code by factoring n_A and then computing the deciphering key (n_A,d_A).

   (c) Explain why, even without factoring n_A, a codebreaker could find the deciphering key rather quickly. In other words, why (in addition to its small size) is 2047 a particularly bad choice for n_A?