The Proof for the Method for Transforming the Modular Inversion Algorithms, Which Are Based on the Extended Euclidean Algorithm, into the Algorithms for the Modular Inversion for the Numbers in the Montgomery Form

Aleksei Vambol
June 2023

Lemma. Let us have an algorithm with the following properties:
- It takes x and n as inputs, where x and n are coprime positive integers;
- It starts with defining its mutable variables as follows: a := x, b := n, u := 1, v := 0;
- It does not use x after defining a;
- It consists of operation blocks, before and after which the mutable variables satisfy the following:
(1) GCD(a, b) = GCD(x, n) = 1,
(2) a = u * x (mod n),
(3) b = v * x (mod n),
(4) a and b are non-negative;
- Before returning the results, it iteratively performs the operation blocks, which decrease a + b, until either a or b is 0, but not both of them, which is impossible for the algorithm;
- It returns u mod n, if b is 0, and v mod n, if a is 0.
Such an algorithm returns the positive multiplicative inverse of x modulo n, and the returned value is less than n.

Proof. If before the returning b is 0, than a is 1 due to (1). Thus, u is the inverse of x modulo n due to (2). The returned value is u mod n, so in this case the algorithm behaves in accordance with the lemma. If before the returning a is 0, than b is 1 due to (1). Thus, v is the inverse of x modulo n due to (3). The returned value is v mod n, so in this case the algorithm behaves in accordance with the lemma. ■

Theorem. Let us have an algorithm for computing the multiplicative inverse of x modulo n, which is based on the Extended Euclidean Algorithm. Since such an algorithm is the particular case of the algorithms described in the lemma, let us abstractly describe this algorithm in accordance with it. *If we initialize u with r mod n instead of 1 at the beginning, where r is a positive number coprime with n, this new algorithm will return the positive multiplicative inverse of x * r^{-1} modulo n, and the returned value will be less than n.*

Proof. Consider the initial algorithm modification obtained by inserting the operation block "a := a * r mod n, u: = u * r mod n" just after "a := x, b := n, u := 1, v := 0". Since GCD(r, n) = 1, the inserted block preserves (1)-(4), so the modified algorithm is also the particular case of algorithms described in the lemma. Thus, if it is executed for the input pair (x * r^{-1} mod n, n), then will return the positive multiplicative inverse of x * r^{-1} modulo n, and the returned value will be less than n. The state of the mutable variables after executing the inserted block can be described in the following way: a = x, b = n, u = r mod n, v = 0. On the other hand, if we execute the new algorithm defined in theorem for the input pair (x, n), then after "a := x, b := n, u := r mod n, v := 0" the state of its mutable variables will be the same as for the modified one. The remaining part of the new algorithm after "a := x, b := n, u := r mod n, v := 0" is also the same as the part of the modified algorithm after the inserted block. Also these remaining algorithm parts do not use the first argument. Thus, the remaining parts of both of these algorithms are the same and operating on the same data, which means that the new algorithm also returns the positive multiplicative inverse of x * r^{-1} modulo n, and the returned value will be less than n. ■

Thus, if the Montgomery representation of y is m = y * R mod N, then this representation for the multiplicative inverse of y modulo N is y^{-1} * R mod N and can be found as the result of executing *the new algorithm*, which is described in the *theorem*, for the input pair (m, N) and r = R^2.