Modular Addition Operation

The modular addition problem is defined as the computation of $S = A + B \pmod{n}$ given the integers A, B, and n. It is usually assumed that A and B are positive integers with $0 \leq A, B < n$, i.e., they are least positives residues. The most common method of computing S is as follows:

1. First compute $S' = A + B$.
2. Then compute $S'' = S' - n$.
3. If $S'' \geq 0$, then $S = S''$ else $S = S'$.

Thus, in addition to the availability of a regular adder, we need fast sign detection which is easy for the CPA, but somewhat harder for the CSA. However, when a CSA is used, the first two steps of the above algorithm can be combined, in other words, $S' = A + B$ and $S'' = A + B - n$ can be computed at the same time. Then, we perform a sign detection to decide whether to take S' or S'' as the correct sum. We will review algorithms of this type when we study modular multiplication algorithms.

Omura's Method

An efficient method computing the modular addition, which especially useful for multioperand modular addition was proposed by Omura in [1]. Let $n < 2^k$. This method allows a temporary value to grow larger than n, however, it is always kept less than 2^k. Whenever it exceeds 2^k, the carry-out is ignored and a correction is performed. The correction factor is $m = 2^k - n$, which is precomputed and saved in a register. Thus, Omura's method performs the following steps given the integers $A, B < 2^k$ (but they can be larger than n).

1. First compute $S' = A + B$.
2. If there is a carry-out (of the kth bit), then $S = S' + m$, else $S = S'$.

The correctness of Omura's algorithm follows from the observations that

- If there is no carry-out, then $S = A + B$ is returned. The sum S is less than 2^k, but may be larger than n. In a future computation, it will be brought below n if necessary.
- If there is a carry-out, then we ignore the carry-out, which means we compute

$$S' = A + B - 2^k \ .$$

The result, which needs to be reduced modulo n, is in effect reduced modulo 2^k. We correct the result by adding m back to it, and thus, compute

$$\begin{aligned} S &= S' + m \\ &= A + B - 2^k + m \\ &= A + B - 2^k + 2^k - n \\ &= A + B - n \ . \end{aligned}$$

After all additions are completed, a final result is reduced modulo n by using the standard technique. As an example, let assume $n = 39$. Thus, we have $m = 2^6 - 39 = 25 = (011001)$. The modular addition of $A = 40$ and $B = 30$ is performed using Omura's method as follows:

$$\begin{array}{lrrll} A &= & 40 = & (101000) & \\ B &= & 30 = & (011110) & \\ S' &= & A + B = & 1(000110) & \text{Carry-out} \\ m &= & & (011001) & \\ S &= & S' + m = & (011111) & \text{Correction} \end{array}$$

Thus, we obtain the result as $S = (011111) = 31$ which is equal to 70 (mod 39) as required. On the other hand, the addition of $A = 23$ by $B = 26$ is performed as

$$\begin{array}{lrrll} A &= & 23 = & (010111) & \\ B &= & 26 = & (011010) & \\ S' &= & A + B = & 0(110001) & \text{No carry-out} \\ S &= & S' = & (110001) & \end{array}$$

This leaves the result as $S = (110001) = 49$ which is larger than the modulus 39. It will be reduced in a further step of the multioperand modulo addition. After all additions are completed, a final negative result can be corrected by adding m to it. For example, we correct the above result $S = (110001)$ as follows:

$$\begin{array}{lrrl} S &= & & (110001) \\ m &= & & (011001) \\ S &= & S + m = & 1(001010) \\ S &= & & (001010) \end{array}$$

The result obtained is $S = (001010) = 10$, which is equal to 49 modulo 39, as required.

References

[1] J. K. Omura. A public key cell design for smart card chips. In *International Symposium on Information Theory and its Applications*, pages 983–985, Hawaii, USA, November 27-30, 1990.