Manuel Leele

We are asked to define an algorithm in order to multiply 2 multiprecision integers a and b

ALGORITHM

Inputs: Multiprecision integers $a = (-1)^s \sum_{0 \le i \le n} a_i 2^{64i}$, $b = (-1)^t \sum_{0 \le j \le m} b_j 2^{64j}$

Output: Multiprecission integer $r = ab$.

Assumption: Our machine contain a subroutine able to perform the multiplication of two single precision integers a and b between 0 an $2^{64}-1$. The output of that subroutine that we are going to denote as spm(a,b) outputs a double precision integer $ab \in \{0, \ldots, 2^{128} - 2^{65} + 1$ so we assume that it returns c, d 64 bits words such that $a \cdot b = d 2^{64} + c$. So $spm(a,b) := (d, c)$ satisfying the latter property

Main Algorithm

for $i = 0, \ldots, n$ do:
    $(d_1, c_0) \leftarrow spm(a_i, b_0)$
    $\gamma_j \leftarrow 0$
    for $j = 1, \ldots, m$ do:
        $(d_{j+1}, c_j) \leftarrow spm(a_i, b_j)$
        $s_j \leftarrow d_j + c_j + \gamma_j$
        $\gamma_{j+1} \leftarrow 0$
        if $s_j \geq 2^{64}$ then:
            $s_j \leftarrow s_j - 2^{64}$
            $\gamma_{j+1} \leftarrow 1$
    $s_{m+1} \leftarrow d_{m+1} + \gamma_{m+1}$
    $r_i \leftarrow (-1)^t \sum_{0 \le j \le m+1} s_j 2^{64j}$
return $r = (-1)^{s+t} \sum_{0 \le i \le n} r_i$

Complexity of the algorithm

We assume that the subroutine spm(a,b) is done in a constant number of machine cycles, same as for summing single precission integers and for the carry flags. Therefore, let n and m be the corresponding lengths of a and b respectively, first of all, per each of the words in a, we are doing m single precision multiplications, single precision sums and carry flags additions, thus Cm arithmetic operations for C constant. Each of those Cm arithmetic operations are done n times 1 per each of the 64-bit words representing a.
Hence, the complexity is $O(nm)$.

//