Exercises left in Magistral Classes

Leonardo Bocchi

February 19, 2023

Exercise 1

Consider the algorithm for the multiplication of multiprecision integers.

Input: Multiprecision integers $a = (-1)^s \sum_{0 \le i \le n} a_i 2^{64i}$, $b = (-1)^t \sum_{0 \le i \le m} b_i 2^{64i}$, not necessarily in standard representation, with $s, t \in \{0, 1\}$.
Output: The multiprecision integer ab.
1. **for** $i = 0, ..., n$ **do** $d_i \leftarrow a_i 2^{64i} \cdot |b|$
2. **return** $c = (-1)^{s+t} \sum_{0 \le i \le n} d_i$

The algorithm computes the product of two single precision integers a, b between 0 and $2^{64} - 1$ which has a "double precision": it lies in the interval $0, ..., 2^{128} - 2^{65} + 1$. We assume the processor has a single precision multiplication instruction that returns the product in two 64-bit words c, d such that $a \cdot b = d \cdot 2^{64} + c$.
Then the algorithm, similarly to the one used for the multiplication of two polynomials, takes the following number of R-operations.

· $(n+1) \times (m+1)$ multiplications (a has n+1 terms and b has m+1 terms). Each term d_i for $i = 0, ..., n$ requires $O(m)$ basic operations
· Summed in $(n + m + 1)$ columns which results in $(n + 1)(m + 1) - (n + m + 1) = nm$ additions

So the total cost of the algorithm is $2nm + n + m + 1 \le 2(n + 1)(m + 1)$, resulting in a complexity of $O(nm)$.