# HOMEWORK 1

Due: 27.02

## 1 Integer division

Let $\beta = 2^w$ be the machine word; in the sequel, we shall assume that the processor, given as input $u \in [0, \beta^2 - 1]$ and $v \in [1, \beta - 1]$, can compute $\lfloor u/v \rfloor$ (the integer part of $u/v$).

Let $A = \sum_{i=0}^{n} a_i \beta^i$ and $B = \sum_{i=0}^{m} b_i \beta^i$ be the expansion of two positive integers in base $\beta$. We will prove that the following algorithm returns $A/B$ and $A \mod B$.

In the algorithm INTEGER DIVISION the values $(a_i), n$ are assumed to be consistent with the updates of $A$. I.e., whenever $A \leftarrow \ldots$, these values are updated.

Integer division

```
procedure GUESS(A, B)
    g ← ⌊(a_n β + a_{n−1})/b_m⌋
    return min(g, β − 1)
end procedure

procedure DIVIDE(A, B)
    while A ≥ B do
        q_{n−m−1} ← GUESS(A, B)
        A ← A − q_{n−m−1} β^{n−m−1} B
        while A < 0 do                          ▷ Correction loop
            q_{n−m−1} ← q_{n−m−1} − 1
            A ← A + B β^{n−m−1}
        end while
    end while
    return (Σ q_k β^k, A)
end procedure
```

1. Prove that, up to multiplying $A$ and $B$ by suitable powers of 2, we can assume that $b_m \geq \beta/2$

2. Prove that, up to replacing $A$ by $A - \beta^{n-m}B$, we can assume that $A < \beta^{n-m}B$.

   Remark: in the previous two questions, also explain how one should correct quotient and remainder in the end to get the result of the original division...

   Define the following loop invariant (for the outer while loop) ; we denote by $A_k, Q_k$ the value of $A, Q$ after the $k$-th iteration of the loop (where $Q$ is the integer $\sum_i q_i \beta^i$ for all $q_i$'s that have been defined so far).

- $0 \leq A_k < \beta^{n-m-k}B$
- $A_k + Q_k B = A$.

where $n$ and $m$ are the maximal powers of $\beta$ that appear in the original $A$ and $B$ (i.e. before the first loop).

3. Deduce that, if the loop invariant is correct, when we exit the loop $(Q, A)$ are the quotient and the remainder of the Euclidean division of $A$ by $B$.

4. Prove the second statement of the loop invariant, and the non-negativity part of the first one.

   We now turn to proving the loop invariant by induction – to simplify the notations we only deal with the first iteration, but the proof carries over to any iteration. We shall split the proof into two cases: either $g \leq \beta - 1$ in GUESS, or GUESS returns $\beta - 1$.

5. First subcase: $g \leq \beta - 1$. Prove that $q_{n-m-1}b_m \geq a_n\beta + a_{n-1} - b_m + 1$. Deduce that

$$A - q_{n-m-1}B\beta^{n-m-1} \leq \sum_{k=0}^{n-2} a_k\beta^k + (b_m - 1)\beta^{n-1},$$

   and the last part of the loop invariant.

6. Second subcase: $g \geq \beta$; prove the last part of the loop invariant *(hint: use question 2)*.

   We now estimate the number of iterations of the Correction loop.

7. Prove that we always have $\text{GUESS}(A, B)b_m \leq a_n\beta + a_{n-1}$, and deduce that

$$A - \text{GUESS}(A, B)\beta^{n-m-1}B > -\text{GUESS}(A, B)\beta^{n-1}.$$

   Deduce that the number of iterations of the Correction loop is at most 2 *(hint: use question 1)*.

8. Conclude on the correction and complexity of the algorithm.

## 2 Multiplication of two polynomials

Give an algorithm to multiply a degree 1 polynomial by a degree 2 polynomial in at most 4 multiplications.

# 3 Composition of polynomials

**A remark for the future:** this exercise is not particularly well formulated. Many are getting confused and do not come up with the right algorithm: students are hinted at splitting $g$ and try to divide-and-conquer on $g$, not on $f$. They do not realize that $m$ remains fixed until the end of the analysis and is optimized for only after $C(n, N)$ is deduced.

1. What is the cost of computing the coefficients of the composition $f \circ g$ of polynomials $f, g$ of degrees $d_1, d_2$? (Assume that ring operations have unit cost.) Use that $f(x) = \sum_{i=0}^{d_1} a_i x^i = a_0 + x(a_1 + x(a_2 + \cdots + x(a_{d_1-1} + a_{d_1} x)\ldots)$.

   Let $N > 0$ be a power of 2 and let $A$ and $B$ be two polynomials over $\mathbb{K}$ with $B(0) = 0$ and $B'(0) \neq 0$. We will study a fast algorithm for computing the composition $A(B) \mod X^N$ which is due to Brent and Kung (1978).

   Let $m > 0$ be a parameter which we will tune later. The algorithm is based on the following Taylor's expansion.

2. Writing $B = B_1 + X^m B_2$ where $B_1$ is a polynomial of degree $< m$ in $\mathbb{K}[X]$, show that

$$A(B) \;=\; A(B_1) \;+\; A'(B_1) X^m B_2 \;+\; A''(B_1)\frac{X^{2m} B_2^2}{2} \;+\; A^{(3)}(B_1)\frac{X^{3m} B_2^3}{6} \;+\; \cdots$$

   Having this decomposition, let us now observe that once we have computed the composition $A(B_1)$ we can compute the other terms efficiently.

3. Let $F$ and $G$ be two polynomials with $G'(0) \neq 0$, and assume that we have computed $F(G) \mod X^N$. Show how to compute $F'(G) \mod X^N$ using $\mathcal{O}(\mathrm{M}(N))$ operations in $\mathbb{K}$, where $\mathrm{M}(n)$ stands for the complexity of multiplying two polynomials of degree $n$ over $\mathbb{K}$.

4. Denoting by $\mathcal{C}(m, N)$ the number of operations used for computing $A(B_1) \mod X^N$, deduce from the previous question a cost bound for computing $A(B) \mod X^N$.

   To obtain a fast algorithm, it remains to give an efficient method to compute $A(B_1) \mod X^N$. To this end, we will study the following more general situation.

5. Let $F$ and $G$ be polynomials over $\mathbb{K}$ of degrees $k$ and $m$ respectively, with $G(0) = 0$. Give a divide-and-conquer algorithm which computes $F(G) \mod X^N$ using $\mathcal{O}(\frac{km}{N}\mathrm{M}(N)\log(N))$ operations in $\mathbb{K}$.

6. Deduce an upper bound for $\mathcal{C}(m, N)$, and a cost bound for computing $A(B) \mod X^N$. Conclude by giving the whole algorithm, including a good choice of $m$ and the corresponding cost bound.