IMPERATIVE PROGRAMMING 2013

MOISE GABRIEL

QUESTION 1

We are given two arrays $xs[0..N)$ and $bs[0..N)$ such that

$$(\forall) i \in [0..N) \cdot 0 \leq xs(i) < bs(i), \quad (1)$$

with

$$xs \otimes bs \triangleq \sum_{j=0}^{N-1} \left(xs(i) \cdot \prod_{j=0}^{i} bs(j) \right)$$

(a)

```
// Function that, given xs and bs, returns xs ⊗ bs, using (N-1) multiplications
def digsToInt(xs: Array[Int], bs: Array[Int]): Int = {
    var N = xs.size
    var i = N-1
    var res = xs(N-1)
    // Invariant I: res = Σ_{j=i}^{N-1} (xs(j)· ∏ bs[i..j)) and we have performed N-1-i
multiplications so far
    // Variant i
    while (i > 0)
    { // I
      res = res * bs(i-1)
      // res = Σ_{j=i}^{N-1} (xs(j)· ∏ bs[(i-1)..j)) and we have performed N-i multiplications
      res = res + xs(i-1)
      // res = Σ_{j=i-1}^{N-1} (xs(j)· ∏ bs[(i-1)..j)) and we have performed N-i multiplications
      i -= 1
      // I
    }
  // I holds && i=0 => res = Σ_{j=0}^{N-1} (xs(j)· ∏ bs[0..j)) and we have performed
N-1 multiplications
  res
}
```

(b) After each iteration of a while loop we will be in the situation:

$$xs(i) + xs(i+1)\cdot bs(i) + xs(i+2)\cdot bs(i)\cdot bs(i+1) + \dots + xs(N-1)\cdot bs(i)\cdot\ldots\cdot bs(N-2) = m_i$$

which is equivalent to

$$xs(i) + bs(i)\cdot\left(xs(i+1) + xs(i+2)\cdot bs(i+1) + \dots + xs(N-1)\cdot bs(i+1)\cdot\ldots\cdot bs(N-2)\right) = m_i$$

By using ①, we get that $(xs(i) < bs(i))$

$xs(i) = m_i \% bs(i)$ and therefore we calculate $xs(i)$

$xs(i+1) + xs(i+2)\cdot bs(i+1) + \dots + xs(N-1)\cdot bs(i+1)\cdot\ldots\cdot bs(N-2) = m_i / bs(i)$ and this is the next step of the iteration.

```
def intToDigs (m: Int, bs: Array[Int]): Array[Int] = {
    var N = bs.size
    var xs = new Array[Int](N)
    var res = m
    var i = 0
    // Invariant I: So far, we have calculated xs[0..i) and res = Σ_{j=i}^{N-1} (xs(j) · Π bs[i..j))
    // Variant N-i
    while (i < N)
    { // I
       xs(i) = res % bs(i) // We know that bs(i) > 0 from ①
       // So far, we have calculated xs[0..(i+1)) and res = Σ_{j=i}^{N-1} (xs(j) · Π bs[i..j))
       res = res / bs(i)
       // So far, we have calculated xs[0..(i+1)) and res = Σ_{j=i+1}^{N-1} (xs(j) · Π bs[(i+1)..j))
       i += 1
       // I
    }
    // I holds && i = N => we have calculated xs[0..N), so we can return it
    xs
}
```

At each iteration we need one division and one modulo operation, so we needed $2N$ operations of this kind.