COL 226 - Assignment 1

Harikesh (2019CS10355)

January 22, 2023

1 Assumptions/Decisions

1. In case when the input is an empty string (""), I have outputted the pair of quotient and remainder as ("" , "") i.e. both of them are assumed as empty strings.
2. In case when the remainder has one or more zeroes (0's) in the start, I have returned the remainder by striping the remainder string for 0's in the starting. For example, when the remainder is "00000" it is returned as "0" and when the remainder is "0064", it is returned as "64".
3. The above assumption is taken care of at each point of the problem i.e., all remainders, and that thing is also taken care of in the case of quotients as well as dividends (initially stripping).
4. I have assumed that the string given has to contain only digits and if any character in the input string is other than the digit form of character i.e. s[i]<0 or s[i]>9 then we will return the error as a pair i.e. ("The input given is wrong", "Please correct it!!!").

2 Pseudo code

Now, for writing the pseudo-code of the algorithm we will be given a string let's say s and then we have to return the pair of the quotient and remainder let's say that any time the quotient and the remainder are q and r. Now we will be defining a new function isqrtld which takes input as s and return (q,r) pair finally. Here I have assumed that for our algorithm we are given the two operations on the strings i.e. stringmult() and stringsub() where stringmult is the function for multiplying a string of any length to a string of length 1 and stringsub is a function for subtracting two strings recursively. Also, assume that we have the inbuilt/ coded algorithm for stripping a string for 0's in the starting.

Algorithm 1 Finding the square root of a string by using long division method

Require: String s, stripped, stringmult, stringsub functions
  Check for basic ”” and non-digit strings recursively
  return (””, ””) and (”The input given is wrong”, ”Please correct it!!!”) resp.
  $s \leftarrow stripped(s)$
  $recsqrt(s, q, r, ind)$
  $len \leftarrow length(s)$

  Now in recsqrt at any time
  if $ind = len$ **then**
    return (q, r)
  else if (len-ind) is odd **then**
    $divisor \leftarrow stringmult(q, 2) \oplus (suitable\_digit)$
    $r \leftarrow (r \oplus s[ind]) - divisor \times (suitable\_digit)$
    $q \leftarrow q \oplus (suitable\_digit)$
    recursively call recsqrt(s, q, r , ind + 1)
  else
    $divisor \leftarrow stringmult(q, 2) \oplus (suitable\_digit)$
    $r \leftarrow (r \oplus s[ind : ind + 2]) - divisor \times (suitable\_digit)$
    $q \leftarrow q \oplus (suitable\_digit)$
    recursively call recsqrt(s, q, r , ind + 2)
  end if

  $suitable\_digit$(number, tobefound, divisor)
  Now in recursive $suitable\_digit$ function
  $final\_divisor \leftarrow$ divisor $\oplus$ number
  $final\_product \leftarrow final\_divisor \times$ number
  if $final\_product > tobefound$ **then**
    return number - 1
  else
    recursively call $suitable\_digit$(number+1, tobefound, divisor)
  end if

3 Proof of Correctness

We will be proving this algorithm by using induction. We are given that we have some large integer/string whose digits are $a_{n-1}, \ldots, a_0$ and n>0. Then we have to show that using the above algorithm, and we will get the correct quotient and remainder. We will be inducting on the number of pairs we will have until that point, indirectly the induction on the length of the string (digits in the number). The proof of correctness goes as follows:

3.1 Base Case

For the base case, we have the integer/string given to us will be of length 1 or 2 i.e., the digits in a are a_0 or a_1a_0. Then, we can say that using the algorithm we have, we will be outputting (b, r) where b is such that $b^2 \leq a < (b+1)^2$ and also r is the remainder i.e., r = a - b^2. This is because we will have only one pair, and initially, both our divisor and quotient are empty, and thus we will be choosing some number b such that $b^2 \leq a < (b+1)^2$ and the remaining thing will be our remainder which is a - b^2.

3.2 Inductive Hypothesis

Now, we assume that the above algorithm gives the correct result till m pairs and till processing m pairs, we have the quotient as the square root of the number formed only by digits $a^{'} = a_{n-1}, \ldots, a_{n-2m}$ (even if we assume the odd number of digits that will have very much less effect on this) and let the quotient be $b^{'}$ till now and remainder at this point is $r^{'}$. Then using our definition of square root i.e., our algorithm, we have that $b^{'2} \leq a^{'} < (b^{'}+1)^2$ and also $r^{'}$ = $a^{'}$ - $b^{'2}$.

Now, we will say that we will move to the next step in our recursion, i.e. $(m+1)^{th}$ pair. Now, let's say we took down the pair $a_{n-2m-1}a_{n-2m-2}$ to the remainder, and thus we have thus the new number, i.e., the dividend at this point formed is 100 $\times$ $r^{'}$ + 10 $\times$ a_{n-2m-1} + a_{n-2m-2}.
Now, according to our algorithm, we have our divisor as double the quotient in the previous step added to some maximum integer (let i) such that the product of the divisor and that integer is less than the new dividend i.e.

(20 $\times$ $b^{'}$ + i) $\times$ i $\leq$ 100 $\times$ $r^{'}$ + 10 $\times$ a_{n-2m-1} + a_{n-2m-2} < (20 $\times$ $b^{'}$ + (i +1)) $\times$ (i + 1)

Because we took the largest such i while using our algorithm.
Now, on putting $r^{'}$ as $a^{'}$ - $b^{'2}$ we get

($20b^{'}$i + i^2) $\leq$ $100a^{'}$ - $100b^{'2}$ + $10a_{n-2m-1}$ + a_{n-2m-2} < ($20b^{'}$ + (i +1)) $\times$ (i + 1) which on simplification i.e., taking $100{b^{'}}^2$ to the other sides give us:

$$(10b^{'} + i)^2 \leq 100a^{'} + 10a_{n-2m-1} + a_{n-2m-2} < (10b^{'} + i + 1)^2$$

Now, we can see that $100a^{'} + 10a_{n-2m-1} + a_{n-2m-2}$ is nothing but the number from the start containing m+1 pairs which let's say for now is a_{new}. Also, we can see that $(10b^{'} + i)$ is our new quotient, and let's say it is b_{new}. Thus, we can now see that we have got a new number b_{new} s.t.

${b_{new}}^2 \leq a_{new} < (b_{new} + 1)^2$ and also r_{new} now will be $a_{new} - {b_{new}}^2$.

Thus, we can say that if our algorithm is true for m pairs, it will also be true for (m+1) pairs, which was initially true in the case of a single pair.
Since the induction hypothesis holds along with the base case of the algorithm, we can say that the above algorithm we have implemented is also correct, i.e., it gives the quotient and remainder pair correctly.

Now, we have to show that apart from the above algorithm, the other trivial algorithms we have used are also correct. The first one of them is the subtraction of two strings recursively. That is true because at any point in time, we are taking any particular index in the two strings and carry over either 0 or -1, depending on the situation (compared on the carry + larger string index to the smaller string index), and the index is moved to index+1.

The other one of them is the multiplication of two strings recursively. That is true because we take any particular index in the larger string at any point in time and carry over either 0 or 1, depending on the situation (after multiplication). The index is moved to index+1.

Thus, we can say that the pseudo-code/ algorithm we have implemented is correct because it gives us the quotient and remainder pair such that $q^2 \leq s < (q+1)^2$ and s - q^2 holds.