Big Multiplication - Algorithm Analysis 406

Daus Carmichael

May 2021

1 Addition

Given two arrays of digits each with a value: $0 < d < 2^3 2 - 1$ show the addition algorithm is correct by proving the statement

$$c + \sum_{k<i} as[k] = \sum_{k<i} (as'[k] + bs[k]$$

is true at the top of the loop for every iteration.

[Basis]

$$c_0 + \sum_{k<i} as[k] = \sum_{k<i} (as'[k] + bs[k])$$

$0 = 0$, as the carry value is initialized to 0 and the sums have no terms for $i = 0$ thus the initial statement is true

[Inductive Hypothesis]

$$c + \sum_{k<i} as[k] = \sum_{k<i} (as'[k] + bs[k])$$

[Inductive Step]

$$c + \sum_{k<i+1} as[k] = \sum_{k<i+1} (as'[k] + bs[k])$$

$$c + as[i+1] + \sum_{k<i} as[k] = as'[i] + bs[i] + \sum_{k<i} (as'[k] + bs[k])$$

$$as[i+1] = as'[i+1] + bs[i+1]$$

by subtracting the inductive hypotheses from both sides and the lines of code

```
s = (uint64_t) as[i] + (uint64_t) bs[i] + (uint64_t) c;
c = s >> 32;
as[i] = (uint32_t) s;
```

show that the next term is

$$c + as[i+1] = as'[i+1] + bs[i]$$

.

2 Partial Product

2.1 Algorithm

In decimal multiplication, each digit is multiplied by the entire list of digits and these are summed together. Shown in the example below, the purpose of the algorithm "partialprod32" is to do this:

ab

·c

$$a * c * B^1 + b * c * B^0$$

2.2 Carry propagation

The code has a carry bit that may populate as[i+2] where i+2 may be greater sz_a. To ensure that the algorithm is valid this must be not lead to a loss of data.

For a number of length $\boldsymbol{n}$ digits of 32 bits, the largest possible value is $2^{32n} - 1$ and the largest value of the second number is $2^{32} - 1$. The largest resulting value from this is:

$$= 2^{32n} * 2^{32} - 2^{32} - 2^{32n} + 1$$

$$= 2^{32(n+1)} - 2^{32n} - 2^{32} + 1$$

Which is clearly within the bound $p < 2^{32(n+1)} - 1$.

2.3 Proof of correctness

[Basis]

$$\sum_{k<0} as[k] = c \cdot B^{i+2} + \sum_{k<0} bs[i] * d$$

[Inductive Hypothesis]

$$\sum_{k<i} as[k] = c \cdot B^{i+2} + \sum_{k<i} bs[k] * d$$

[Inductive Step]

$$\sum_{k<i+1} as[k] = c \cdot B^{i+1+2} + \sum_{k<i+1} bs[k] * d$$

By subtracting the inductive hypothesis, as[i+1] = c + bs[i+1]*d as c is on the same order now as the largest part of the product could be

```
as[i] = as[i-1]+bs[i]*d;
as[i+1] = as[i]+bs[i]*d;
as[i+2] += c;
```

Matching what was given although the sizes of the 2 are different, in the proof, as[i+1] must be greater than the size of bs[i] by 1 'digit.'