Integer Squaring

Sepand Haghighi
Mohammad Abassi

Winter - 2017

Outline

- Introduction
- Integer Multiplication
- Integer Squaring
- Error
- Implementation

Introduction

$$c = a \times b \bmod p\,.$$

- Integer multiplication, and
- Integer modular reduction.

$$a = (a_{n-1}, \ldots, a_2, a_1, a_0)_\beta\,.$$

$$(\varepsilon_{i+1}, c_i) = a_i + b_i + \varepsilon_i, \qquad i = 0, 1, 2, \ldots$$

Integer Multipication

Algorithm 1: Integer multiplication (by operand scanning)

INPUT: n-digit integers a and b.
OUTPUT: $2n$-digit integer $d = a \times b$.

1. **for** $i = 0$ **to** $n-1$ **do** $d_i = 0$
2. **for** $i = 0$ **to** $n-1$ **do**
3. $\quad H = 0$
4. $\quad A = a_i$
5. $\quad$ **for** $j = 0$ **to** $n-1$ **do**
6. $\quad\quad (H, L) = A \times b_j + H + d_{i+j}$
7. $\quad\quad d_{i+j} = L$
8. $\quad d_{i+n} = H$
9. **return**(d)

$$d = a \times b = \sum_{i=0}^{n-1} a_i b\, \beta^i.$$

Integer Multipication

Algorithm 2: Integer multiplication (by product scanning)

INPUT: n-digit integers a and b.
OUTPUT: $2n$-digit integer $d = a \times b$.

1. $(U,H,L) = (0,0,0)$
2. **for** $k = 0$ **to** $2n-2$ **do**
3. **if** $k < n$ $I = \{i \mid 0 \le i \le k\}$
4. **if** $k \ge n$ $I = \{i \mid n > i > k-n\}$
5. **for** every $i \in I$
6. $(U,H,L) \mathrel{+}= a_i \times b_{k-i}$
7. $d_k = L$
8. $(U,H,L) = (0,U,H)$
9. $d_{2n-1} = L$
10. **return**(d)

$$d = \sum_{k=0}^{2n-2} \beta^k \left(\sum_{i \in I} a_i b_{k-i} \right), \qquad I = \{i \mid 0 \le i, k-i < n\}.$$

Integer Squaring

Algorithm 3: Integer squaring

INPUT: n-digit integer a.
OUTPUT: $2n$-digit integer $d = a^2$.

1. $(U,H,L) = (0,0,0)$
2. **for** $k = 0$ **to** $2n-2$ **do**
3. **if** $k < n$ $I = \{i \mid 0 \le i < k/2\}$
4. **if** $k \ge n$ $I = \{i \mid n > i > k/2\}$
5. **for** every $i \in I$
6. $(U,H,L) \mathrel{+}= a_i \times a_{k-i}$
7. **if** k is even $(U,H,L) = 2(U,H,L) + a_{k/2}^2$
8. **if** k is odd $(U,H,L) = 2(U,H,L)$
9. $d_k = L$
10. $(U,H,L) = (0,U,H)$
11. $d_{2n-1} = L$
12. **return**(d)

$$\sum_{\substack{i\in I \\ i\neq k-i}} a_i a_{k-i} = 2 \sum_{\substack{i\in I \\ i>k-i}} a_i a_{k-i} = 2 \sum_{\substack{i\in I \\ i<k-i}} a_i a_{k-i}$$

Error!!

Algorithm 3: Integer squaring

INPUT: n-digit integer a.
OUTPUT: $2n$-digit integer $d = a^2$.

1. $(U,H,L) = (0,0,0)$
2. **for** $k = 0$ **to** $2n-2$ **do**
3. **if** $k < n$ $I = \{i \mid 0 \le i < k/2\}$
4. **if** $k \ge n$ $I = \{i \mid n > i > k/2\}$
5. **for** every $i \in I$
6. $(U,H,L) \mathrel{+}= a_i \times a_{k-i}$
7. **if** k is even $(U,H,L) = 2(U,H,L) + a_{k/2}^2$
8. **if** k is odd $(U,H,L) = 2(U,H,L)$
9. $d_k = L$
10. $(U,H,L) = (0,U,H)$
11. $d_{2n-1} = L$
12. **return**(d)

$(U,H,L) = (0,0,0)$
$(U,H,L)' = (0,0,0)$

$(U,H,L) = (U,H,L) + (U,H,L)'$
$d_k = L$
$(U,H,L)' = (0,U,H)$
$(U,H,L) = (0,0,0)$

Implementation(IntegerSquaring)

```python
def IntegerSquaring(Input_integer,Base=10):
    '''
    Integer Squaring Algorithm
    Book : Cryptographic Engineering (Serdar S¨ uer Erdem, Tu˘ grul Yanık, and C ¸ etin Kaya Koc)
    Chapter : 5
    Page : 80
    :param Input_integer : Input Integer as string with space separation between digit "0 11 12" in base 13
    :type Input_integer : str
    '''
    try:
        IntegerList=list(map(int,list(Input_integer.split(" "))))
        for number in IntegerList:
            if number>Base-1:
                raise Exception("[Erro] Bad Number In This Base")
        IntegerList.reverse()
        IntegerLength=len(IntegerList)
        I=[]
        d=[]
        UHL=0
        UHLPrev=0
        for k in range((2*IntegerLength-2)+1):
            if k<IntegerLength:
                I=list(range(int(math.ceil(k/2))))
            else:
                I=list(range((k//2)+1,IntegerLength))
            for i in I:
                UHL=UHL+IntegerList[i]*IntegerList[k-i]
            if (k%2)==0:
                UHL=2*UHL+IntegerList[int(k/2)]**2
            else:
                UHL=2*UHL
            UHL=UHL+UHLPrev
            L=UHL%Base
            U=UHL//(Base**2)
            H=(UHL-U*(Base**2)-L)//Base
            d.append(L)
            L=H
            H=U
            U=0
            UHLPrev=H*Base+L
            UHL=0
        if L!=0:
            d.append(L)
        d.reverse()
        d=list(map(str,d))
        return " ".join(d)
    except Exception as e:
        print(str(e))
        return 0
```

https://github.com/sepandhaghighi/Integer-Squaring

Implementation(IntegerMulti.)

```
def IntegerMultiplication(Input_integer_1,Input_integer_2,Base=10):
    '''
    Integer Squaring Algorithm
    Book : Cryptographic Engineering (Serdar S¨ uer Erdem, Tu˘ grul Yanık, and C ¸ etin Kaya Koc)
    Chapter : 5
    Page : 79
    :param Input_integer_1 : Input Integer as string with space separation between digit "0 11 12" in base 13
    :param Input_integer_2 : Input Integer as string with space separation between digit "0 11 12" in base 13
    :type Input_integer_1 : str
    :type Input_integer_2 : str
    '''
    IntegerList1 = list(map(int, list(Input_integer_1.split(" "))))
    IntegerList2 = list(map(int, list(Input_integer_2.split(" "))))
    IntegerList1.reverse()
    IntegerList2.reverse()
    IntegerLength = max(len(IntegerList1),len(IntegerList2))
    IntegerList1.extend([0]*(IntegerLength-len(IntegerList1)))
    IntegerList2.extend([0] * (IntegerLength - len(IntegerList2)))
    I = []
    UHL=0
    d=[]
    for k in range((2*IntegerLength-2)+1):
        if k<IntegerLength:
            I = list(range(k+1))
        else:
            I=list(range(k-IntegerLength+1,IntegerLength))
        for i in I:
            UHL = UHL + IntegerList1[i] * IntegerList2[k - i]
        L = UHL % Base
        U = UHL // Base**2
        H = (UHL - U * (Base**2) - L) // Base
        d.append(L)
        L = H
        H = U
        U = 0
        UHL = H*Base+L
    if L!=0:
        d.append(L)
    d.reverse()
    d = list(map(str, d))
    return " ".join(d)
```

https://github.com/sepandhaghighi/Integer-Squaring

Thanks ;-)