Algoritmo veloce di elevazione a potenza modulo n ($B^m \mod n$)

October 17, 2022

1 rappresentazione dell'esponente come somme e prodotti

Supponiamo di voler calcolare $B^m \mod n$.
Sia:

$$m = (b_0 b_1 b_2 \cdots b_n)_2$$

(dove i b_i sono cifre binarie e b_0 è la cifra meno significativa)

$$m = b_0 + 2(b_1 + 2(b_2 + 2(\cdots(b_n + 2 \cdot 1))))$$

Esempio:

$$m = 13_{10} = (1101)_2 = 1 + \overbrace{2(0 + \underbrace{2(1 + 2 \cdot 1)}_{6})}^{12} \tag{1}$$

Ed allora:

$$B^{b_0+2(b_1+2(b_2+2(\cdots(b_n+2\cdot 1))))} \tag{2}$$

Osservando la (2) si può notare come partendo dal bit meno significativo b_0 dell'esponente l'algoritmo può essere riassunto nei seguenti passi:

1. si parte dal bit meno significativo b_0 e si procede fino a b_n
2. se $b_i = 0$ si raddoppia l'esponente (cioè si eleva B al quadrato)
3. se $b_i = 1$ si raddoppia e si somma 1 (oltre ad elevare B al quadrato si moltiplica il risultato B)

Questo perché dalla (2) posso scrivere:

$$B^{b_0+2(b_1+2(b_2+2(\cdots(b_n+2\cdot 1))))} = B^{b_0}(B^{b_1}(B^{b_2}(\cdots(B^{b_n})^2)^2)^2)^2 \tag{3}$$

Esempi:

$$3^{12} = 3^{(1100)_2} = 3^{0\overbrace{(3^0(\underbrace{3^1(3^1)^2}_{3\times(3)^2=27})^2)^2}^{(27^2)^2=531441}} \tag{4}$$

$$5^{20} = 5^{(10100)_2} = 5^0 \underbrace{(5^0 \overbrace{(5^1(\underbrace{5^0(5^1)^2}_{625})^2)}^{(5\times625)^2=9765625})^2}_{(9765625)^2=95367431640625})^2 = (((5(5)^2)^2)^2)^2 = (5)^4 \times (5^{16})$$

2 Algoritmo

```
#Algoritmo di conversione da decimale a binario
def dec2bin(n):
        b=[ ]
        while n>0:
                b.insert(0,'0') if (n%2==0) else b.insert(0,'1')
                n= int (n/2)
        return b
#potenza con metodo delle quadrature ripetute
def potqr(b,exp,mod):
        c,d=0,1
        esp2=dec2bin(exp) #converte in binario l'esponente
        for k in range (0, len (esp2)):
                d=(d * d)%mod
                if esp2[k]=="1":
                        d=(d * b)%mod
        return d
```