IIC3253

Teoría de números

Para recordar: aritmética modular

Dados $a, n \in \mathbb{Z}$, existe un único par de elementos $(q, r) \in \mathbb{Z}^2$ tal que:

$$0 \leq r < |n|$$

$$a = q \cdot n + r$$

Cuociente

Resto

Decimos entonces que $a \bmod n = r$

Para recordar: aritmética modular

Además, $a \equiv b \mod n$ si y solo si $n|(b-a)$

- Lo cual es equivalente a $a \bmod n = b \bmod n$

Una propiedad fundamental: si $a \equiv b \mod n$ y $c \equiv d \mod n$, entonces

$$(a+c) \equiv (b+d) \mod n$$

$$(a \cdot c) \equiv (b \cdot d) \mod n$$

Nuestro objetivo inicial

Vamos a estudiar algunos algoritmos en teoría de números

- Los cuales son fundamentales para los protocolos criptográficos de clave pública

Máximo común divisor

Sean $a, b \in \mathbb{N}$ con $a > b$. Para calcular $MCD(a, b)$ utilizamos la siguiente recurrencia:

$$MCD(a,b) = \begin{cases} a & \text{si } b = 0 \\ MCD(b, a \bmod b) & \text{si } b > 0 \end{cases}$$

¿Cómo se transforma esto en un algoritmo? ¿Cuál es su complejidad?

Una nota sobre la complejidad

En los protocolos para criptografía de llave pública podemos usar números con 800 dígitos

Si n tiene 800 dígitos, no vamos a poder ejecutar un algoritmo de tiempo polinomial en n

- No podemos realizar n, n^2 o n^3 operaciones, puesto que $n \geq 10^{799}$

Una nota sobre la complejidad

Vamos a utilizar algoritmos de tiempo polinomial en el largo de n

- Vale decir, de tiempo polinomial en el largo de la entrada

Esto significa que vamos a utilizar algoritmos de tiempo polinomial en $\log(n)$

El algoritmo extendido de Euclides

Dados números $a, b \in \mathbb{N}$ tales que $a > b$, existen números $s, t \in \mathbb{Z}$ tales que:

$$MCD(a, b) = s \cdot a + t \cdot b$$

Por ejemplo: $MCD(180, 63) = 9$ y $9 = (-1) \cdot 180 + 3 \cdot 63$

Dados números a y b, el algoritmo extendido de Euclides calcula $MCD(a, b)$ y los números s, t

El algoritmo extendido de Euclides

Defina una secuencia $r_0, r_1, \ldots$ tal que:

$$r_0 = a$$

$$r_1 = b$$

$$r_{i+1} = r_{i-1} \bmod r_i \qquad \text{para } i \geq 1$$

Si $r_k = 0$, entonces $MCD(a, b) = r_{k-1}$ y el algoritmo se puede detener

El algoritmo extendido de Euclides

Además, vamos a mantener secuencias de números s_0, $s_1, \ldots$ y $t_0, t_1, \ldots$ tales que

$$r_i = s_i \cdot a + t_i \cdot b$$

Si $r_k = 0$, entonces $MCD(a,b) = r_{k-1} = s_{k-1} \cdot a + t_{k-1} \cdot b$ y el algoritmo retorna $MCD(a,b)$, s_{k-1} y t_{k-1}

El algoritmo extendido de Euclides

Tenemos que:

$$r_0 = 1 \cdot a + 0 \cdot b$$

$$r_1 = 0 \cdot a + 1 \cdot b$$

$$r_{i+1} = \left(s_{i-1} - \lfloor \tfrac{r_{i-1}}{r_i} \rfloor \cdot s_i\right) \cdot a + \left(t_{i-1} - \lfloor \tfrac{r_{i-1}}{r_i} \rfloor \cdot t_i\right) \cdot b$$

Esto se deduce considerando que $r_{i+1} = r_{i-1} \bmod r_i$ y:

$$r_{i-1} \ = \ \lfloor \frac{r_{i-1}}{r_i} \rfloor \cdot r_i \ + \ r_{i-1} \bmod r_i$$

Un ejemplo del algoritmo extendido de Euclides

$$180 = 1 \cdot 180 + 0 \cdot 63$$

$$63 = 0 \cdot 180 + 1 \cdot 63$$

$$180 \bmod 63 = 54$$

Un ejemplo del algoritmo extendido de Euclides

$$180 = 1 \cdot 180 + 0 \cdot 63$$

$$63 = 0 \cdot 180 + 1 \cdot 63$$

$$54 =$$

Un ejemplo del algoritmo extendido de Euclides

$$180 = 1 \cdot 180 + 0 \cdot 63$$

$$63 = 0 \cdot 180 + 1 \cdot 63$$

$$54 =$$

$$s_2 = 1 - \lfloor \tfrac{180}{63} \rfloor \cdot 0$$

Un ejemplo del algoritmo extendido de Euclides

$$180 = 1 \cdot 180 + 0 \cdot 63$$

$$63 = 0 \cdot 180 + 1 \cdot 63$$

$$54 = 1 \cdot 180 +$$

Un ejemplo del algoritmo extendido de Euclides

$$180 = 1 \cdot 180 + 0 \cdot 63$$

$$63 = 0 \cdot 180 + 1 \cdot 63$$

$$54 = 1 \cdot 180 +$$

$$t_2 = 0 - \lfloor \tfrac{180}{63} \rfloor \cdot 1$$

Un ejemplo del algoritmo extendido de Euclides

$$180 = 1 \cdot 180 + 0 \cdot 63$$

$$63 = 0 \cdot 180 + 1 \cdot 63$$

$$54 = 1 \cdot 180 + (-2) \cdot 63$$

$$9 = (-1) \cdot 180 + 3 \cdot 63$$

$$0 = 7 \cdot 180 + (-20) \cdot 63$$

¿Cuál es la complejidad del algoritmo extendido de Euclides?

Demuestre que este algoritmo funciona en tiempo polinomial en el largo de la entrada

Inverso modular

b es inverso de a en módulo n si:

$$a \cdot b \equiv 1 \mod n$$

Por ejemplo, 4 es inverso de 2 en módulo 7:

$$2 \cdot 4 \equiv 1 \mod 7$$

¿Cuándo un número es invertible en módulo n?

El número 2 no es invertible en módulo 8

Teorema: un número a es invertible en módulo n si y solo si $MCD(a, n) = 1$

- Vale decir, si a y n son primos relativos

¿Cómo calculamos el inverso modular?

Dados números a y n:

1. Verifique que $MCD(a, n) = 1$
2. Si el paso 1. se cumple, use el algoritmo extendido de Euclides para construir $s, t \in \mathbb{Z}$:
$$1 = MCD(a, n) = s \cdot a + t \cdot n$$
3. Retorne s

Un ejemplo del cálculo de inverso modular

Tenemos que $MCD(140, 33) = 1$, por lo que 33 tiene inverso en módulo 140

Utilizando el algoritmos extendido de Euclides obtenemos que:

$$1 = MCD(140, 33) = (-4) \cdot 140 + 17 \cdot 33$$

Por lo tanto: 17 es inverso de 33 en módulo 140

Un ejemplo del cálculo de inverso modular

Dado que $1 = (-4) \cdot 140 + 17 \cdot 33$, también tenemos que -4 es inverso de 140 en módulo 33

Y dado que $-4 \equiv 29 \mod 33$, concluimos que 29 es inverso de 140 en módulo 33

Exponenciación rápida

No podemos calcular a^b si a y b son números de 800 dígitos

- El resultado tiene demasiados dígitos

Pero sí podemos calcular $a^b \bmod n$, ya que este número está entre 0 y $n-1$

¿Cómo hacemos esto? Veamos esto en la pizarra