Matemática Discreta

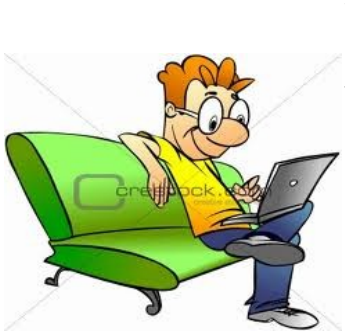
Cálculo de las claves del algoritmo RSA

Sesión de laboratorio

Laburpena

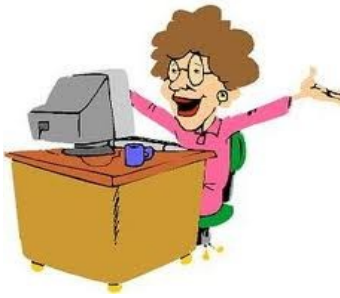
Se pretende implementar en R el conjunto de funciones que calculen las claves del algoritmo RSA. Dicho algoritmo utiliza una clave pública y otra privada para cifrar y descifrar los mensajes. La pública la componen dos números n y r, mientras que la privada consta de un único número s.

Criptografía:
Algoritmo de cifrado RSA

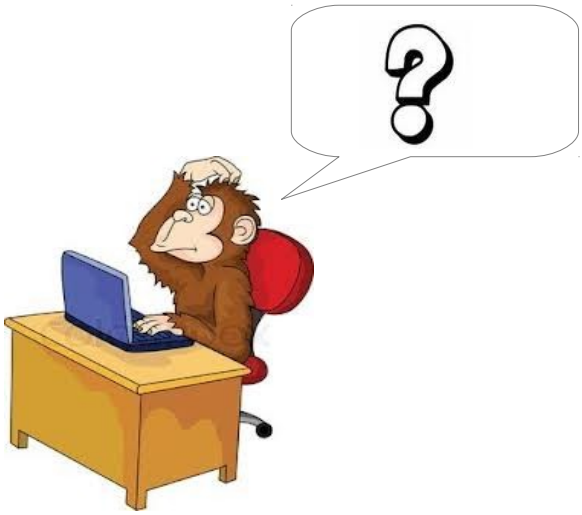


Te digo un secreto?

Si! RSA, ASCII, n=391, r=3



RSA, ASCII, 40, 79, 265, 171, 304



Matemática Discreta
Ana Zelaia Jauregi

1 Las claves del algoritmo RSA

- La clave pública (n, r). Quien quiera recibir un mensaje cifrado genera la clave pública y se lo hace saber a todo el mundo, a cualquiera que pretenda enviarle mensajes cifrados. Da igual que quien quiera interceptar el mensaje conozca los números n y r que componen la clave, ya que para descifrar el mensaje hace falta la clave privada. La clave pública se utiliza para cifrar el código que representa el mensaje.

- Clave privada (s). Se trata del número que hace falta para descifrar los mensajes cifrados mediante la clave pública formada por n y r. Sin conocer s dificilmente se descifrará el mensaje. Es por ello que haya que guardarla en secreto (en privado), para que solamente la persona a quien va dirigido el mensaje sea capaz de descifrarlo.

Para los cálculos necesarios en la obtención de las claves haremos uso de las funciones de R: `Primes(a,b)`, `nextPrime(a)`, `previousPrime(a)`, `modinv(a,b)`, `extGCD(a,b)`, `primeFactors(a)`, `format(a, scientific=FALSE)`.

2 Ejercicios

1. Define la función llamada `claves_RSA(p,q)` que, dados los números primos p y q, calcule n, r y s. Los pasos a seguir son:

> 1. Elegir dos números primos p y q, con $p \neq q$.
> 2. Calcular $n = p \times q$.
> 3. Calcular un primo relativo r de $m = (p-1) \times (q-1)$, es decir, r debe cumplir $\text{mcd}(m, r) = 1$. La clave publica es (n, r).
> 4. La clave privada s. Hay que calcular el inverso módulo m del valor r, es decir, $s = r^{-1} \bmod m$.

   En el tercer paso del algoritmo podemos utilizar la función `primo_relativo_minimo(m)` implementada en la sesión de laboratorio anterior. Con la función `claves_RSA(p,q)` se obtendrán las siguientes claves para cada siguiente par de números primos:

   - claves_RSA(5,17): Clave pública, n= 85 r= 3, privada, s= 43
   - claves_RSA(17,23): Clave pública, n= 391 r= 3, privada, s= 235
     (Cálculo manual en las hojas de teoría sobre el algoritmo de cifrado RSA)
   - claves_RSA(97,101): Clave pública, n= 9797 r= 7, privada, s= 2743
   - claves_RSA(307,397): Clave pública, n= 121879 r= 5, privada, s= 96941

   Desgraciadamente, las claves que se han obtenido no son nada seguras. Si las utilizáramos nos romperían el sistema facilmente, ya que la descomposición de los números n de las claves es factible en muy poco tiempo, y, una vez descompuesto el valor n, cualquiera podría calcular el valor m necesario para el cálculo de s. Y si conocemos s estamos en la misma situación que la persona a la que va dirigida el mensaje, ya que tendríamos la clave secreta s. Demuestra que utilizando R puedes calcular la clave privada a partir de las públicas.

   Elige dos números primos mayores, puedes mirar en la lista de números primos de wikipedia:

   https://es.wikipedia.org/wiki/Anexo:Números_primos

   Las claves son para ti, las eliges tu... ¿Son seguras?

2. Implementa la función `claves_RSA_grandes(a,b)`, que basándose en dos números grandes $a, b \in \mathbb{Z}$ calcule las claves pública y privada. A la función implementada en el ejercicio anterior, `claves_RSA(p,q)` cámbiale lo siguiente:

   - Parámetros: La función recibirá dos números enteros positivos, no tienen por qué ser primos. Será la propia función la que se encargue de calcular los números primos p y q, utilizando la función `nextPrime`. De esta forma facilitamos el uso de la función a quien quiera calcular sus propias claves, ya que no le pedimos dos números primos iniciales.

- Claves: queremos que tanto n como r sean numeros grandes, para dificultar la descomposición de n en factores. Podemos utilizar la función de la sesión anterior `primo_relativo(m,t)` que calcula un primo relativo mayor que t, siendo t un número tan grande como queramos.
- Escritura científica: En R los números grandes se muestran según la escritura científica, que oculta cifras y utiliza exponentes. Pero queremos ver todos los dígitos de los números que componen las claves, Para que R muestre todos los dígitos de un número, le podemos forzar a que lo haga anulando la opción de que muestre los números con el formato científico: `format(a, scientific=FALSE)`.

Utiliza la nueva función con los siguientes números para obtener las claves:

```
claves_RSA_grandes(2634758697353,293756536383)
```

Trata de obtener la clave privada s a partir de la clave pública... ¿Son seguras las nuevas claves? ¿Por qué? ¿Donde está el secreto de la seguridad? Puedes realizar más pruebas, con números iniciales más pequeños y tratar de obtener la clave privada a partir de la pública.

Un artículo sobre el tema: Competición de factorización RSA

https://es.wikipedia.org/wiki/Competición_de_factorización_RSA