Exponenciador Cuaternario

Zelzin Marcela Márquez Navarrete

Departamento de Computación, CINVESTAV

Exponenciador Cuaternario

Se realizó la implementación del algoritmo cuaternario para la exponenciación de un número con exponente de 32 bits en C++.

# Exponenciación modular

Al realizar un exponenciación modular se debe recordar que no se debe calcular:

ya que hacer esto representa un gasto muy grande de memoria por la forma en la que crece el valor exponenciado, por lo que el resultado debe ser reducido en cada ocasión para poder ser almacenado.

## Exponenciación binaria

El método binario revisa bit a bit de el exponente y se realiza una elevación al cuadrado en cada paso y dependiendo del valor del bit evaluado se realiza una multiplicación.

### Exponenciación cuaternaria.

El método binario puede ser generalizado tomando los bits de a 2, 3, 4, …, etc., para la exponenciación cuaternaria tomamos los bits de dos en dos. Preprocesando los valores para poder acceder a ellos cuando se lean de los bits del exponente como se muestra en la tabla 1.

Tabla 1. Valores preprocesados para la exponenciación cuaternaria

|  |  |  |
| --- | --- | --- |
| **Bits** | **W** | **M^w** |
| **00** | **0** | **1** |
| **01** | **1** | **M** |
| **10** | **2** | **M^2** |
| **11** | **3** | **M^3** |

#### Implementación.

El código cuenta con tres funciones:

*void* preprocessing(*unsigned* *long* M, *unsigned* *long*\* calculatedExp);

/\*\*

Returns the index values to access the preproceesed array.

Access by pairs the F array in which are stored the bits of exponent e.

@param e The exponent.

@param index The index to be acessed in the preprocessed array.

\*/

*unsigned* *long* get\_Fi(*unsigned* *long* e, *unsigned* index);

/\*\*

Returns M^e (mod n).

@param M The radix.

@param e The exponent.

@param n The divisor.

\*/

*unsigned* *long* quaternaryExp(*unsigned* *long* M, *unsigned* *long* e, *unsigned* *long* n);

##### Preprocessing.

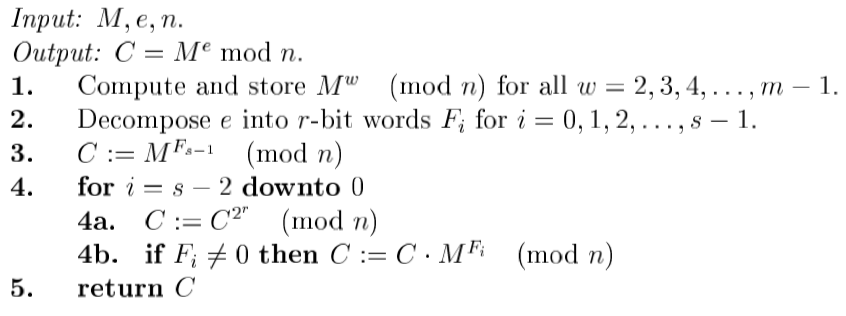
Realiza el precálculo mencionado en la tabla 1.

##### Get\_Fi

Permite acceder de forma constante al valor de los dos bits a leer del exponente con los cuales se acedera al valor correspondiente del arreglo preprocesado. Esto se logra enmascarando los bits del exponente mediante una ‘*and*’ entre el exponente *e* y un 3 recorrido el número de veces que indique el índice a ser accedido, de 0 a 15, ya que 32/2 es 16, con 32 el número de bits del exponente y 2 el número de palabra *w* por ser cuaternario y esto por 2 veces el índice.

##### QuaternaryExp

De izquierda a derecha realiza la revisión de cada uno de los bits del exponente para realizar la elevación al cuadrado y una multiplicación en caso de que el bit sea 1, de acuerdo con este algoritmo, tomado de [1].



Donde *M* es la base, *e* el exponente, *n* el divisor, *r* es igual a 2 y *Fi* es el valor accedido del par de bits de *e*.

Referencias

1. High-Speed RSA Implementation. Çetin Kaya Koç. RSA Laboratories