UNIVERSIDAD NACIONAL DE INGENIERÍA
FACULTAD DE CIENCIAS
ESCUELA PROFESIONAL DE MATEMÁTICA

TESIS PARA OPTAR EL TÍTULO PROFESIONAL DE
LICENCIADO EN MATEMÁTICAS

TITULADA:

ALGORITMOS PARA LA DETERMINACIÓN DE PRIMALIDAD DE NÚMEROS ENTEROS

PRESENTADA POR:

OSWALDO JOSÉ VELÁSQUEZ CASTAÑÓN

LIMA-PERÚ

2003

Dedicada a la memoria de mi
abuelo, José Velásquez Flores.

Agradezco a mi asesor, el Dr. Julio Alcántara, por todas las molestias que se tomó, a mis padres Oswaldo y Sara por su apoyo incondicional y a Yboon por la insistencia.

Contenido

Introducción

El problema de distinguir los números primos de los compuestos es un problema fundamental de las matemáticas. El primer algoritmo conocido que trabaja correctamente hallando los números primos, la Criba de Eratóstenes, data del 240 A.C. En el siglo XVII, Fermat probó el resultado conocido como el *Pequeño Teorema de Fermat.* Aunque de simple formulación, ese resultado fue la base para la creación de algoritmos modernos para determinar primalidad

El nacimiento de la era de la computación permitió, gracias a la enorme cantidad de datos que pueden procesar en poco tiempo las computadoras, el trabajar con números de un tamaño mucho mayor al posible utilizando tan sólo herramientas como lápiz y papel. La necesidad de medir de forma teórica la velocidad de los programas de las máquinas llevó al desarrollo de la teoría de la complejidad computacional. A partir de ahí, todos los algoritmos a ser implementados se midieron en función de dicha teoría. Dado un entero n, se utiliza como parámetro para medir la velocidad de un algoritmo la función logaritmo $\log n$. Llamamos *algoritmo polinomial* a un algoritmo tal que el número de pasos para su ejecución crece a lo más tan rápido como un polinomio en $\log n$, éstos son considerados los mejores algoritmos; si cambiamos el polinomio por una función exponencial, tenemos lo que se conoce como *algoritmo exponencial*, un ejemplo lo da la criba de Eratóstenes. A pesar de los avances con respecto al problema de determinación de primalidad, no fue hasta 1978 que el problema tomó mayor relevancia, con la creación del RSA (por las iniciales de sus autores, Rivest, Shamir y Adleman), el método de codificación de mensajes (encriptación) más utilizado en Internet, inclusive hasta ahora. Esto despertó el interés de la industria por el problema de primalidad, ya que las claves del método se generan precisamente a partir de números primos.

Con el nuevo enfoque del problema, surgió una interrogante: ¿existe un algoritmo para determinar primalidad en tiempo polinomial? Podemos hacer un recuento sobre los esfuerzos con respecto a este problema. A partir del pequeño teorema de Fermat, se creó un test probabilístico para determinar si un número dado es compuesto. A pesar de que este test falla para los llamados *números de Carmichael*, sirvió para la creación de algoritmos más eficientes para el trabajo con números primos. En 1976, Miller utilizó estas

propiedade para crear un test determinístico que determina primalidad en tiempo polinomial, pero asumiendo una conjetura de la teoría de números, la *Hipótesis de Riemann Extendida*. Más adelante, en 1980, Rabin modificó este test para crear un test probabilístico para hallar números compuestos, que trabaja en tiempo polinomial. Sin embargo este algoritmo no puede determinar efectivamente si un número entero es primo. Un test probabilístico similar que trabaja en tiempo polinomial fue dado en 1977 por Solovay y Strassen.

No fue hasta 1983 cuando Adleman, Pomerance y Rumely dieron el primer algoritmo determinístico que prueba efectivamente primalidad en un tiempo de ejecución menor que exponencial, $(\log n)^{c \log\log\log n}$ (para una constante c). Luego, en 1986, Goldwasser y Killian dieron un algoritmo basado en la teoría de Curvas Elípticas, que determina primalidad en tiempo polinomial en un subconjunto "denso" de enteros (en un sentido especial), aunque no sobre todos). Más tarde, en 1992, Fellows y Koblitz probaron que si la factorización de $n-1$ es conocida, entonces se puede determinar en tiempo polinomial la primalidad del entero n determinísticamente. Números de forma especial, como los *números de Mersenne*, pueden ser también verificados en tiempo polinomial, determinísticamente (test de Lucas y Lehmer 1935). Sin embargo se hacía muy difícil hallar un algoritmo de propósito general para probar primalidad en tiempo polinomial; siempre se sacrificaba, o bien velocidad, o bien generalidad. Fue finalmente en agosto del 2002 cuando Agrawal, Kayal y Saxena hallaron un algoritmo con estas características. El desarrollo del algoritmo requiere sólo herramientas básicas de teoría de números, a excepción de un resultado de la teoría de cribas (cuya prueba no es nada elemental).

En este trabajo, se reúnen varios algoritmos de los mencionados. Hay que resaltar que sólo se han reunido algoritmos que requieren herramientas básicas de la teoría de números para su comprobación, las cuales son dadas en el capítulo 1. Se adopta un enfoque moderno, considerando la teoría de complejidad de algoritmos; y aunque la meta de este trabajo no es la implementación de los algoritmos, la notación de pseudocódigo utilizada vuelve muy fácil esta tarea, en un lenguaje de programación estructurado tal como *C* o *Pascal*, o un lenguaje como *Mathematica*. En el capítulo 2 se presentan algoritmos probabilísticos que sirven para determinar si un número es compuesto con certeza, aunque si determinan que un número dado es posiblemente primo, ello ocurre salvo por una pequeña probabilidad de error. Luego, en el capítulo 3 se dan algunos algoritmos que permiten probar incondicionalmente si un número es primo, pero sólo si este número obedece a una forma especial. El capítulo 4 está dedicado al algoritmo de Agrawal-Kayal-Saxena que, como mencionamos, es el primer algoritmo que determina en tiempo polinomial si un número dado cualquiera es primo o compuesto. Finalmente, en el capítulo 5 se estudia el criptosistema RSA y los posibles ataques al mismo.

Capítulo 1

Preliminares

En este capítulo desarrollamos las herramientas básicas necesarias para el trabajo en los siguientes capítulos, éstos son conceptos de la teoría algebraica y analítica de números y de la teoría de complejidad computacional. El uso en la práctica de algunas de las herramientas requiere del desarrollo de algoritmos, los cuales presentaremos aquí. Suponemos que el lector se encuentra familiarizado con las propiedades básicas de los conjuntos de números reales ($\mathbb{R}$) y enteros ($\mathbb{Z}$), la programación básica en un computador y la notación de pseudocódigo, y algunas nociones básicas de la teoría de probabilidades.

1.1 Números enteros

El lector debe estar familiarizado con el principio de inducción, una herramienta básica para el desarrollo de la teoría:

Principio de Inducción. Si Q es un conjunto de enteros tales que

1. $k \in Q$, y
2. $n \in Q$ implica $n + 1 \in Q$,

entonces todo entero $\geq k$ pertenece a Q.

Existe otro principio, lógicamente equivalente al principio de inducción, que también será utilizado en adelante:

Principio de Buena Ordenación. Si A es un conjunto no vacío de enteros acotado inferiormente (esto es, existe $k \in \mathbb{Z}$ tal que todo elemento de A es $\geq k$), entonces A posee un elemento mínimo.

Definición 1.1.1 Sean n, d números enteros. Decimos que d *divide a* n y lo denotamos

por $d|n$ si $n = cd$ para algún $c \in \mathbb{Z}$. Diremos que n es un *múltiplo de* d, que d es un divisor de n o que d es un factor de n. Si d no divide a n, denotamos $d \not| n$.

Teorema 1.1.1 (Propiedades de la divisibilidad) Se cumplen, para $a, b, c, x, y \in \mathbb{Z}$,

1. si $a|b$ y $b|c$, entonces $a|c$ (transitividad),
2. si $a|b$ y $a|c$, entonces $a|bx + cy$ (linealidad),
3. si $a|n$, entonces $ax|nx$,
4. si $ax|nx$ y $x \neq 0$, entonces $a|n$,
5. si $d|n$ y $n \neq 0$, entonces $|d| \leq |n|$,
6. si $d|n$ y $d \neq 0$, entonces $\frac{n}{d}|n$.

Teorema 1.1.2 (El algoritmo de división) Dados dos enteros a y b con $b > 0$, existe un único par de enteros q y r tales que

$$a = bq + r, \qquad \text{con } 0 \leq r < b.$$

Además, $r = 0$ si y sólo si $b|a$.

Prueba. Sea S el conjunto de enteros no negativos dado por

$$S = \{y : y = a - bx,\ x \in \mathbb{Z},\ y \geq 0\}.$$

Si $a \geq 0$, entonces $a = a - b \cdot 0 \in S$; si $a < 0$ entonces $a - ba = a(1 - b) \in S$. En cualquier caso S es no vacío. Por el principio de la buena ordenación, S admite un elemento mínimo $r = a - bq$. Como $r - b < r$ pero $r - b = a - b(q + 1)$, entonces $r - b < 0$ pues de otro modo $r - b \in S$ contradiciendo la minimalidad de r. Por lo tanto

$$0 \leq r < b.$$

Veamos ahora la unicidad. Supongamos que $a = bq_1 + r_1$ con $0 \leq r_1 < b$, entonces

$$b(q - q_1) = r_1 - r.$$

Como $b|(r_1 - r)$ y $|r_1 - r| < b$, entonces $r_1 - r = 0$ de donde $r_1 = r$ y $q_1 = q$.

Veremos más adelante cómo calcular efectivamente estos números q, r. Esto podrá hacerse conociendo la forma de dividir números más pequeños (cifras). Los enteros q y r

obtenidos del algoritmo de la división aplicado a a y b son llamados *cociente* y *residuo* (o *resto*) respectivamente; y denotamos

$$r = a \bmod b.$$

Observe que, vistos como números reales, $a/b = q + r/b$, con $0 \leq r/b < 1$, y de donde $q = \lfloor a/b \rfloor$, la parte entera de a/b. Por ello en adelante utilizaremos la notación $\lfloor a/b \rfloor$ para indicar la división entera de a y b.

Definición 1.1.2 Un entero n se dice que es *primo* si $n > 1$ y los únicos divisores positivos de n son 1 y n. Si $n > 1$ y n no es primo, decimos que n es *compuesto*.

Ejemplo 1.1.1 Los primeros números primos son

$$2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97.$$

Lema 1.1.1 Todo entero $n > 1$ puede expresarse como producto de primos.

Prueba. Utilizaremos inducción sobre n. Para $n = 2$ el teorema es trivial. Supongamos que el teorema es cierto para todo entero $< n$. Si n es primo, el teorema es válido; de otro modo $n = cd$, donde $1 < c, d < n$. Pero cada uno de ellos es un producto de números primos (por hipótesis inductiva), y por lo tanto lo es n.

Teorema 1.1.3 (Euclides) Existe una infinidad de números primos.

Prueba. Supongamos, por reducción al absurdo, que sólo existe un número finito de primos, $p_1, p_2, \ldots, p_n$. Sea $m = p_1 p_2 \cdots p_n + 1$. Como $m > 1$, o bien m es primo, lo cual es imposible por hipótesis (ya que es mayor que todos los primos), o m es producto de primos. Sin embargo ninguno de los primos p_i puede dividir a m, de lo contrario dividiría a $m - p_1 \cdots p_n = 1$, absurdo.

1.2 Complejidad computacional

En el estudio de los algoritmos de la Teoría de Números, se presentan las preguntas naturales:

- Dado un algoritmo A para resolver un problema en particular, ¿cuándo es éste eficiente? Esto es, se ejecuta con la suficiente rapidez?
- Dados dos algoritmos A y B, ¿cuál de ellos es superior?

- Dado un problema P, ¿existe una solución para el problema? Si bien puede existir una solución teórica, ¿es ésta efectivamente calculable? ¿Hay un número mínimo de pasos que se requieren para resolver el problema?

Estamos interesados en el estudio de métodos para el trabajo con números naturales (o a lo más con enteros). Dado un número natural a, y otro número natural b dado, podemos escribir a como

$$a = \sum_{i=0}^{n} a_i b^i \text{ donde } 0 \leq a_i < b$$

para todo $0 \leq i \leq n$, y $n = \lfloor \log_b a \rfloor$. Diremos que

$$(a_n \ldots a_1 a_0)_b$$

es la *representación de a en la base b*, y $n+1$ es el *número de cifras* (o *dígitos*) en la base b.
Supongamos que existe una máquina que realiza las siguientes operaciones (llamadas *primitivas* o *elementales*) con naturales m y n menores que b:

(C) comparación de m con n;

(S) suma de m con n, dando como resultado la representación de $m+n$ en la base b,

$$(a_1 a_0)_b = m + n;$$

(R) substracción de m de n, en caso que m sea menor o igual a n;

(M) multiplicación de m por n, dando la representación de mn en la base b,

$$(a_1 a_0)_b = m \cdot n;$$

(D) división de un número de dos dígitos en la base b por m, dando como resultado un cociente con dos dígitos en la base b y un resto con un dígito en la base b,

$$(b_1 b_0)_b = (a_1 a_0)_b \cdot m + r, \quad 0 \leq r < m < b.$$

En base a esto, queremos establecer una medida de la eficiencia de un algoritmo. Parece natural medir la eficiencia por el tiempo de ejecución de un algoritmo, lo que a su vez puede ser medido por el número de operaciones elementales realizadas en su ejecución. Entonces se presenta un problema: es posible que para dos algoritmos A y B se tenga que A opera mejor sobre ciertas instancias mientras que B opera mejor sobre otras. Siendo así entonces, ¿cómo uniformizamos el estudio? Podemos interpretar la complejidad del algoritmo en

función del tamaño de la instancia. Aún así, puede ocurrir que sobre instancias del mismo tamaño el desempeño no sea uniforme. Entonces podemos considerar o bien un desempeño promedio sobre las instancias o bien un desempeño en el peor caso. Aquí usaremos este último enfoque (análisis del peor caso), pues un estudio de complejidad promedio requiere el analizar una distribución (probabilística) de las instancias de un algoritmo, lo cual torna el estudio demasiado complejo.
Siendo así, consideremos que la máquina con la que trabajamos realiza las operaciones C, S, R, M y D en el peor de los casos en tiempos $T(C)$, $T(S)$, $T(R)$, $T(M)$ y $T(D)$, respectivamente. Definimos para un algoritmo $\#(X)$ como el número de operaciones del tipo X que éste algoritmo realiza. El *costo* total del algoritmo es

$$\#(C)T(C) + \#(S)T(S) + \#(R)T(R) + \#(M)T(M) + \#(D)T(D).$$

Algoritmo 1.2.1 Dados dos números naturales $n = (n_r \dots n_1 n_0)_b$ y $m = (m_r \dots m_1 m_0)_b$, este algoritmo decide si los números son iguales (podemos considerar el mismo número de cifras, de otro modo completamos uno de ellos poniendo ceros a la izquierda):

Datos:	$n = (n_r \dots n_1 n_0)_b$ $m = (m_r \dots m_1 m_0)_b$
$i = 0$	
mientras $(i \leq r \wedge n_i = m_i)$	
	$i = i + 1$
Salida:	Si $i > r$ entonces "son iguales" sino "son distintos"

Luego el costo del algoritmo es

$$f(r) = (r+1)(2T(C) + T(S)) + T(C).$$

Muchas veces, no es posible hallar exactamente el número de pasos necesarios para resolver una instancia de tamaño r de un algoritmo dado. En vez de ello, nos contentamos con hallar una función f tal que, para alguna constante k y r suficientemente grande, $k \cdot f(r)$ sea una cota superior para el número de pasos necesarios para ejecutar una instancia de tamaño r.

Definición 1.2.1 Considere funciones $f, g : \mathbb{N} \to \mathbb{R}$, con $g(r) \geq 0$ para todo $r \in \mathbb{R}$. Diremos que

$$f(r) = O(g(r))$$

cuando existen constantes $k \in \mathbb{R}^+$ y $N_0 \in \mathbb{N}$ tales que

$$|f(r)| \leq kg(r), \text{ para } r \geq N_0.$$

Con esta definición, decimos que un algoritmo A para resolver un problema P tiene *complejidad* (o *costo*) $O(g(r))$ si el número de pasos $f(r)$ requeridos por el algoritmo para resolver una instancia de tamaño r cumple $f(r) = O(g(r))$.

Ejemplo 1.2.1 El costo del algoritmo 1.2.1 es $O(r)$.

La inclusión de la constante multiplicativa k en la definición tiene varias consecuencias. Una gran ventaja es que ella torna automáticamente equivalentes a dos modelos computacionales en los que las operaciones elementales de uno pueden ser simulados por un número finito de operaciones elementales del otro. Así, cuando decimos que un algoritmo tiene complejidad $O(g(n))$, esto es válido para una amplia familia de modelos computacionales (e independiente de la máquina en la que se trabaje). De este modo, podemos aumentar ciertas primitivas a nuestro modelo (por ejemplo una operación de exponenciación de dígitos E) sin alterar la complejidad del algoritmo.
Entre las desventajas: puede ocurrir que el desempeño de dos algoritmos de la misma complejidad teórica sea muy distinto, ya que la constante k en la definición puede ser arbitrariamente grande. Además, puede ocurrir que nos interese trabajar, en la práctica, con instancias de tamaño $r < N_0$, lo que invalida la cota del número de pasos $k \cdot g(r)$. Por ello la complejidad del algoritmo $O(g(n))$ es llamada *complejidad asintótica.*
Adoptamos este modelo en todo caso, por su simplicidad y amplia difusión. Un "buen" algoritmo es uno en el que el crecimiento del número de pasos de un algoritmo no sea muy rápido. Para nuestros propósitos, buscamos algoritmo de complejidad

$$O(p(r))$$

donde p es un polinomio (*complejidad polinomial, tiempo polinomial*) en el número de dígitos r de los números involucrados en el cálculo.

En algunos casos requerimos calcular la complejidad de un algoritmo que depende de varios datos de entrada independientes. Para esto damos la siguiente definición:

Definición 1.2.2 Sean $f, g : \mathbb{N}^s \to \mathbb{R}$ funciones con $g(r_1, \ldots, r_s) \geq 0$ en $\mathbb{N}^s$. Decimos que

$$f(r_1, \ldots, r_s) = O(g(r_1, \ldots, r_s))$$

si existen constantes $k \in \mathbb{R}^+$ y $N_0 \in \mathbb{N}$ tales que

$$|f(r_1, \ldots, r_s)| \leq k g(r_1, \ldots, r_s), \text{ para } r_j \geq N_0,\ 1 \leq j \leq s.$$

1.3 Aritmética Ilimitada

Nece itamo algoritmo para el trabajo con números enteros grandes; en esta sección e tablecemos las operacione aritméticas básicas para ello y medimos su velocidad.

Algoritmo 1.3.1 Este algoritmo suma dos números enteros dados $m = (m_r \dots m_1 m_0)_b$ y $n = (n_r \dots n_1 n_0)_b$.

Entrada:	$m = (m_r \dots m_1 m_0)_b$ $n = (n_r \dots n_1 n_0)_b$
$k = 0$	
para $i = 0, \dots, r$	
	$(k\, s_i)_b = m_i + n_i + k$
$s_{r+1} = k$	
Salida:	$m + n = (s_{r+1} \dots s_1 s_0)_b$

El costo de este algoritmo es

$$(r+1)(2T(S)) = O(r).$$

Tan importante como la suma de dos números es el calcular su diferencia:

Algoritmo 1.3.2 Este algoritmo calcula la diferencia de dos números naturales dados $m = (m_r \dots m_1 m_0)_b$ y $n = (n_r \dots n_1 n_0)_b$.

Entrada:	$m = (m_r \dots m_1 m_0)_b$ $n = (n_r \dots n_1 n_0)_b$ $m \geq n > 0$
$k = 0$	
para $i = 0, \dots, r$	
	$s_i = (m_i - n_i + c) \bmod b$
	si $(m_i - n_i + c) \geq 0$
	$c = 0$
	sino
	$c = -1$
Salida:	$m - n = (s_r \dots s_1 s_0)_b$

Si deseamos calcular una diferencia cuando $m < n$, entonces debemos realizar la siguiente modificación al algoritmo: al terminar el algoritmo, si $c = -1$, repetimos el mismo algoritmo pero ahora con los datos $m = (00\dots0)_b$ ($r+1$ ceros), $n = (s_r \dots s_1 s_0)_b$.

Algoritmo 1.3.3 Este algoritmo multiplica dos números enteros dados $m = (m_r \dots m_1 m_0)_b$ y $n = (n_r \dots n_1 n_0)_b$.

Entrada:	$m = (m_s \dots m_1 m_0)_b$ $n = (n_r \dots n_1 n_0)_b$
para $i = 0, \dots, s+r+1$	
	$p_i = 0$
para $i = 0, \dots, r$	
	$c = 0$
	para $j = 0, \dots, s$
	$(uv)_b = p_{i+j} + x_j \cdot y_j + c$ $p_{i+j} = v$ $c = u$
	$p_{i+s+1} = u$
Salida:	$m \cdot n = (p_{s+r+1} \dots p_1 p_0)_b$

En caso que $r = s$, tenemos que el costo de este algoritmo es $O(r^2)$.

Algoritmo 1.3.4 Este algoritmo divide dos números enteros dados $m = (m_r \dots m_1 m_0)_b$ y $n = (n_r \dots n_1 n_0)_b$. Para este algoritmo exigimos que $s \geq r \geq 1$, $n_r \neq 0$; el caso general sólo requiere una pequeña modificación.

Entrada:	$m = (m_s \dots m_1 m_0)_b$, $n = (n_r \dots n_1 n_0)_b$ con $s \geq r \geq 1$, $n_r \neq 0$
para $j = 0, \dots, (s-r)$	
	$q_j = 0$
mientras $(m \geq nb^{s-r})$	
	$q_{s-r} = q_{s-r} + 1$ $m = m - nb^{s-r}$
para $i = s, \dots, r+1$ [conteo en descenso]	
	si $(m_i = n_r)$
	$q_{i-r-1} = b - 1$
	sino
	$q_{i-r-1} = \lfloor (m_i b + m_{i-1})/n_r \rfloor$
	mientras $(q_{i-r-1}(n_r b + n_{r-1}) > m_i b^2 + m_{i-1} b + m_{i-2})$
	$q_{i-r-1} = q_{i-r-1} - 1$
	$m = m - q_{i-r-1} n b^{i-r-1}$
	si $(m < 0)$
	$m = m + nb^{i-r-1}$ $q_{i-r-1} = q_{i-r-1} - 1$
$r = m$	
Salida:	cociente $q = \lfloor m/n \rfloor = (q_{s-r} \dots q_1 q_0)_b$, residuo $r = m \bmod n$ que cumplen $m = qn + r$, $0 \leq r < n$

El algoritmo posee tres bucles principales. El primer bucle inicializa el cociente en 0. El segundo se ejecuta a lo más una vez si $n_r \geq \lfloor b/2 \rfloor$ y b es par. El estimado de q_{i-r-1} en la primera parte del tercer bloque del algoritmo nunca es menor al valor real. El siguiente paso, de correción del estimado del algoritmo, no se repite más de dos veces si $n_r \geq \lfloor b/2 \rfloor$. Se puede garantizar que $n_r \geq \lfloor b/2 \rfloor$ multiplicando m, n por un λ conveniente (evidentemente menor que b); este proceso es denominado *normalización*. Si

$$m = qn + r, 0 \leq r \leq n$$

entonces

$$\lambda m = q(\lambda n) + (\lambda r), 0 \leq \lambda r \leq \lambda n$$

Escogiendo λ conveniente (de un dígito), el algoritmo hará el número de pasos previsto, el resultado obtenido del algoritmo se divide por λ. Para esto hemos de implementar el cociente de un número entero cualquiera entre un número de un solo dígito (caso $r = 0$), lo cual es fácil, y toma tiempo lineal $O(s)$.

El algoritmo de división tiene costo total $O(s^2)$. Observe que las multiplicaciones por potencias de b, realizadas en el algoritmo, tienen costo lineal.

Independencia de la base

Una observación importante: supongamos que m, n son dos números de magnitud similar. Sin pérdida de generalidad, podemos suponer que $m \leq n$. Como el número de cifras de n en la base b es $r = \lfloor \log_b n \rfloor + 1$, la suma de m y n tiene complejidad $O(\log n)$ y tanto el producto como la división tienen complejidad $O(\log^2 n)$ pues si $p = p(r)$ es un polinomio en r,

$$O(p(\lfloor \log_b n \rfloor + 1)) = O(p(\log_b n)) = O(p(\log n))$$

donde $a \in \mathbb{R}^+ \setminus \{1\}$ es cualquiera. Esto significa que el costo de las operaciones aritméticas no depende de la base en la que las realizamos.

1.4 El Máximo Común Divisor

Definición 1.4.1 Si un entero d divide a dos enteros a y b, entonces decimos que d es un *divisor común* de a y b. Por ejemplo, $d = 1$ es un divisor común de dos enteros cualesquiera.

Teorema 1.4.1 Dados dos enteros a y b, existe un entero d y sólo uno, tal que

1. $d \geq 0$,
2. $d|a$ y $d|b$,
3. si $e|a$ y $e|b$, entonces $e|d$.

Además d se escribe en la forma

$$ax + by = d$$

donde x, y son enteros. Este número se denomina *máximo común divisor* (m.c.d.) de a y b y se denota por $\text{mcd}(a, b)$ o (a, b)(si no hay peligro de confusión). Si $(a, b) = 1$ entonces se dice que a y b son *primos entre sí* (o *coprimos*).

Prueba. Si $a = 0$, $b = 0$ no hay nada que probar (escogemos $d = 0$). En otro caso, definimos

$$I(a, b) = \{ax + by : x, y \in \mathbb{Z}\}.$$

Escogemos el menor elemento positivo de $I(a, b)$, $d = ax_0 + by_0$. Por el algoritmo de la división, existen enteros q, r con

$$a = dq + r, \quad 0 \leq r < d,$$

entonces

$$r = a - dq = a(1 - qx_0) + b(-qy_0) \in I(a, b),$$

y como $r < d$, entonces $r = 0$ pues de otro modo se contradice la minimalidad de d. Entonces $d|a$. Análogamente, probamos que $d|b$. Sea ahora e entero tal que $e|a$ y $e|b$, entonces

$$e|ax_0 + by_0 = d$$

como se esperaba. Veamos la unicidad de d: Sabemos que si $e|a$ y $e|b$, entonces $e|d$; por otro lado si e posee la propiedad (iii) de la definición de m.c.d., dado que $d|a$ y $d|b$ por (ii), entonces $d|e$. Por lo tanto $d = e$.

Teorema 1.4.2 (Propiedades del m.c.d.) Dados a, b, c, d, n, k enteros, se cumplen

(i) $(a, b) = (b, a)$,

(ii) $(a, (b, c)) = ((a, b), c)$,

(iii) $(ac, bc) = |c|(a, b)$,

(iv) $(a, 1) = (1, a) = 1$, $(a, 0) = (0, a) = |a|$,

(v) si $(a, b) = (a, c) = 1$, entonces $(a, bc) = 1$,

(vi) si $(a, b) = 1$, entonces $(a^n, b^k) = 1$, para cualesquiera $n, k \geq 1$.

¿Cómo calcular el m.c.d. de dos enteros dados? Consideremos dos enteros positivos a, b (ya que $(a, b) = (|a|, |b|)$), y la siguiente secuencia de divisiones:

$$\begin{aligned}
a &= bq_1 + r_1 \quad \text{con} \quad 0 \leq r_1 < b, \\
b &= r_1q_2 + r_2 \quad \text{con} \quad 0 \leq r_2 < r_1, \\
r_1 &= r_2q_3 + r_3 \quad \text{con} \quad 0 \leq r_3 < r_2, \\
r_2 &= r_3q_4 + r_4 \quad \text{con} \quad 0 \leq r_4 < r_3, \\
&\vdots
\end{aligned}$$

Acomodando los restos de las divisiones, tenemos

$$b > r_1 > r_2 > r_3 > r_4 > \cdots \geq 0.$$

Como entre b y 0 sólo hay una cantidad finita de enteros, entonces existirá un resto $r_n = 0$. Si consideramos el primer resto nulo de la secuencia, entonces afirmamos que el último resto no nulo de este proceso, r_{n-1}, es (a, b). La justificación se basa en el siguiente

Lema 1.4.1 Sean a, b enteros positivos. Si existen enteros q, r tales que $a = bq + r$, entonces $(a, b) = (b, r)$.

La demostración del lema es trivial. Ahora consideremos el esquema

$$\begin{array}{lll} a = bq_1 + r_1 & \text{con} & 0 \leq r_1 < b, \\ b = r_1 q_2 + r_2 & \text{con} & 0 \leq r_2 < r_1, \\ r_1 = r_2 q_3 + r_3 & \text{con} & 0 \leq r_3 < r_2, \\ \quad \vdots & & \\ r_{n-3} = r_{n-2} q_{n-1} + r_{n-1} & \text{con} & 0 \leq r_{n-1} < r_{n-2}, \\ r_{n-2} = r_{n-1} q_n & \text{y} & r_n = 0. \end{array}$$

Aplicando repetidamente el lema anterior, tenemos

$$(a, b) = (b, r_1) = (r_1, r_2) = (r_2, r_3) = \cdots = (r_{n-2}, r_{n-1}) = (r_{n-1}, 0) = r_{n-1}.$$

Algoritmo 1.4.1 (Algoritmo Euclidiano) El siguiente algoritmo calcula el m.c.d. de dos enteros a y b dados.

Entrada: a, b enteros	
$r = a$	
mientras $r \neq 0$	
	$r = a \bmod b$ $a = b$ $b = r$
Salida:	(a, b) en la variable a

Teorema 1.4.3 (Lema de Euclides) Sean a, b, c enteros. Si $a|bc$ y $(a, b) = 1$, entonces $a|c$.

Prueba. En efecto, como existen x, y tales que $ax + by = 1$, entonces $acx + bcy = c$. Como además $a|bc$, por la linealidad de la división, $a|c$.

Teorema 1.4.4 Para p primo, a, b enteros

(i) si $p \nmid a$, entonces $(p, a) = 1$,

(ii) si $p|ab$, entonces $p|a$ o $p|b$.

Prueba. Sea $r = (p, a)$, como $r|p$, entonces $r = 1$ o $r = p$. Si $r = p$ entonces $p|a$, absurdo. Esto prueba (i). Si $p|ab$ y $p \nmid a$, de (i), $(p, a) = 1$. Entonces del teorema anterior, $p|b$, lo que prueba (ii).

Teorema 1.4.5 (Teorema Fundamental de la Aritmética) Cada entero $n > 1$ se puede representar como un producto de factores primos de manera única, salvo por el orden de los factores.

Prueba. Sólo nos falta probar la unicidad, la posibilidad de descomposición del número se probó en el lema 1.1.1. Probaremos esto por inducción. Para $n = 2$ es obvio pues 2 es primo. Supongamos que

$$n = p_1 \cdots p_m = q_1 \cdots q_l,$$

con p_i, q_j primos, $p_1 \leq \ldots p_m$, $q_1 \leq \ldots, q_l$. Por el teorema anterior, como $p_1 | q_1 \cdots q_l$, entonces $p_1 | q_i$ para algún i, y como q_i es primo, entonces $p_1 = q_i$ y $p_1 \geq q_1$. Análogamente probamos que $p_1 \leq q_1$, y por lo tanto $p_1 = q_1$. Por hipótesis inductiva,

$$n/p_1 = p_2 \cdots p_m = q_2 \cdots q_l$$

admite factorización única (si $n/p_1 = 1$ entonces $n = p_1$ y el resultado es trivial), de donde $m = l$ y $p_i = q_i$ para todo $1 \leq i \leq m$.

Definición 1.4.2 Sean p primo, n entero tales que $p|n$, y $\alpha \geq 1$ entero. Denotamos $p^\alpha \| n$ si p^α es la mayor potencia de p que divide a n, esto es

$$p^\alpha | n \text{ y } p^{\alpha+1} \nmid n.$$

Proposición 1.4.6 Supongamos que $k|m$ y que existe un primo q tal que $k \nmid \frac{m}{q}$. Entonces si $q^\alpha \| m$, tenemos que $q^\alpha \| k$. En particular si $q^\beta | m$, entonces $q^\beta | k$.

Prueba. Escribimos $m = lk$. Como $q|m$, entonces $q|l$ o $q|k$. Si $q|l$ tenemos que $\frac{m}{q} = \frac{l}{q}k$ pero entonces $k|\frac{m}{q}$, absurdo. Luego $(q, l) = 1$ de donde $(q^\alpha, l) = 1$ y $q^\alpha | k$.

Consideremos los pasos del algoritmo de Euclides aplicado a dos números enteros a y b:

$$\begin{aligned}
a &= bq_1 + r_1 && \text{con} \quad 0 \leq r_1 < b, \\
b &= r_1 q_2 + r_2 && \text{con} \quad 0 \leq r_2 < r_1, \\
r_1 &= r_2 q_3 + r_3 && \text{con} \quad 0 \leq r_3 < r_2, \\
&\vdots \\
r_{n-3} &= r_{n-2} q_{n-1} + r_{n-1} && \text{con} \quad 0 \leq r_{n-1} < r_{n-2}, \\
r_{n-2} &= r_{n-1} q_n && \text{y} \quad r_n = 0.
\end{aligned}$$

Entonces $(a, b) = r_{n-1}$.
Supongamos que, en la iteración i-ésima del algoritmo, podemos expresar r_i como combinación lineal de a y b

$$r_i = ax_i + by_i, \quad 1 \leq i \leq n-1.$$

En particular

$$(a, b) = r_{n-1} = ax_{n-1} + by_{n-1}.$$

¿Cómo calculamos x_i e y_i? Supongamos conocidos, en la i-ésima iteración, los valores $x_{i-2}, x_{i-1}, y_{i-2}$ e y_{i-1}. Como

$$r_{i-2} = r_{i-1}q_i + r_i$$

y

$$r_{i-2} = ax_{i-2} + by_{i-2}, \qquad r_{i-1} = ax_{i-1} + by_{i-1}$$

entonces

$$\begin{aligned} r_i &= r_{i-2} - r_{i-1}q_i \\ &= (ax_{i-2} + by_{i-2}) - (ax_{i-1} + by_{i-1})q_i \\ &= a(x_{i-2} - q_i x_{i-1}) + b(y_{i-2} - q_i y_{i-1}) \end{aligned}$$

de modo que podemos hacer

$$x_i = x_{i-2} - q_i x_{i-1}, \qquad y_i = y_{i-2} - q_i y_{i-1}.$$

Sin embargo, esto sólo funciona cuando $i \geq 3$. ¿Qué hacemos cuando $i < 3$? ¿Será que podemos definir x_{-1}, y_{-1}, x_0, y_0 convenientemente? La respuesta es sí. Para esto basta expresar a y b como si fuesen restos de las divisiones.
Observe que del proceso

$$\begin{aligned} a &= bq_1 + r_1, & r_1 &= ax_1 + by_1, \\ b &= r_1q_2 + r_2, & r_2 &= ax_2 + by_2, \\ r_1 &= r_2q_3 + r_3, & r_3 &= ax_3 + by_3, \\ &\vdots \\ r_{n-3} &= r_{n-2}q_{n-1} + r_{n-1}, & r_{n-1} &= ax_{n-1} + by_{n-1}, \\ r_{n-2} &= r_{n-1}q_n, & r_n &= 0. \end{aligned}$$

Podemos construir la tabla

restos	cocientes	x	y
a	$*$	x_{-1}	y_{-1}
b	$*$	x_0	y_0
r_1	q_1	x_1	y_1
r_2	q_2	x_2	y_2
$\vdots$	$\vdots$	$\vdots$	$\vdots$
r_{n-2}	q_{n-2}	x_{n-2}	y_{n-2}
r_{n-1}	q_{n-1}	x_{n-1}	y_{n-1}

Para completar la tabla, vemos que

$$a = ax_{-1} + by_{-1},$$
$$b = ax_0 + by_0,$$

con

$$x_{-1} = 1, \quad y_{-1} = 0, \quad x_0 = 0, \quad y_0 = 1$$

y listo. Observe que, en la práctica, no es necesario almacenar todos los valores en el algoritmo para calcular $(a, b) = d$ y, más aún, para calcular los valores x_{n-1} e y_{n-1} que cumplen

$$d = ax_{n-1} + by_{n-1}$$

(de hecho, en cada paso del algoritmo sólo requerimos los valores de las dos iteraciones anteriores del algoritmo). Estas observaciones nos llevan a la construcción del siguiente algoritmo.

Algoritmo 1.4.2 (Algoritmo Euclidiano Extendido)

Datos: a, b enteros
$x1 = 1, y1 = 0, x0 = 0, y0 = 1$ $r1 = a,\ r0 = b$
repetir
$q = \lfloor r1/r0 \rfloor$ $r = r1 \bmod r0$ $x = x1 - q \cdot x0$ $y = y1 - q \cdot y0$
$x1 = x0$ $x0 = x$ $y1 = y0$ $y0 = y$ $r1 = r0$ $r0 = r$
mientras $r \neq 0$
Salida: $x_1, y_1, r1$ que cumplen $a \cdot x1 + b \cdot y1 = r1 = (a, b)$

¿Cuántas iteraciones (valor de n) realiza el último algoritmo antes de detenerse? De $r_{i+2} < r_{i+1} < r_i$ tenemos

$$r_i = r_{i+1}q_{i+2} + r_{i+2} \geq r_{i+1} + r_{i+2} > 2r_{i+2},$$

luego $r_{i+2} < r_i/2$ para todo i y

$$1 \leq r_{n-1} < \frac{q_0}{2^{\lfloor (n-1)/2 \rfloor}}, \text{ donde } q_0 = \max\{a, b\}$$

de donde

$$\log q_0 > \lfloor (n-1)/2 \rfloor.$$

El número de iteraciones necesarias entonces para ejecutar los últimos algoritmos es $O(\log q_0)$. Si los algoritmos son aplicados a dos números m y n con $n > m$, se realizan $O(\log n)$ iteraciones. En cada iteración, las operaciones realizadas tienen como mayor costo $O(\log^2 n)$ (los números involucrados x_i, y_i tienen $O(\log n)$ cifras). Luego el costo total del algoritmo es $O(\log^3 n)$ si $n > m$.

El máximo común divisor admite generalizaciones. El máximo común divisor de tres enteros a, b, c se denota (a, b, c) y se define por

$$(a, b, c) = (a, (b, c)).$$

Es fácil ver que $(a, b, c) = ((a, b), c)$ y que por lo tanto el m.c.d. depende únicamente de a, b y c y no del orden en el que se escriben. Análogamente, el m.c.d. de n enteros $a_1, \ldots, a_n$ se define inductivamente por

$$(a_1, \ldots, a_n) = (a_1, (a_2, \ldots, a_n)).$$

Del mismo modo, $d = (a_1, a_2, \ldots, a_n)$ es independiente del orden en el que aparecen los a_i. Además, d divide a cada uno de los a_i y que cada divisor común divide a d.

Asociado al máximo común divisor de dos enteros, está el mínimo común múltiplo. Un *múltiplo común* de dos números a y b es un entero d tal que d es múltiplo de a y de b, esto es, $a|d$ y $b|d$. Tenemos el siguiente teorema:

Teorema 1.4.7 Dados dos enteros a y b, existe un entero d y sólo uno, tal que

(i) $d \geq 0$,

(ii) $a|d$ y $b|d$,

(iii) si $a|e$ y $b|e$, entonces $d|e$.

Este número se denomina *mínimo común múltiplo* (m.c.m.) de a y b y se denota por $\text{mcm}(a, b)$.

Es fácil comprobar que este número es nada menos que

$$\text{mcm}(a, b) = \begin{cases} |ab|/(a, b), & \text{si } a \neq 0 \text{ y } b \neq 0, \\ 0 & \text{si } a = 0 \text{ o } b = 0. \end{cases}$$

Ahora podemos definir el m.c.m. de n números $a_1, \ldots, a_n$ inductivamente por

$$\text{mcm}(a_1, a_2, \ldots, a_n) = \text{mcm}(a_1, \text{mcm}(a_2, \ldots, a_n)),$$

de manera análoga al m.c.d.; $d = \text{mcm}(a_1, a_2, \ldots, a_n)$ es un múltiplo común de los a_i y todo múltiplo común de los a_i es múltiplo de d.

1.5 Congruencias

Muchos problemas de la teoría de números que tienen que ver con la divisibilidad de números enteros, pueden simplificarse con la ayuda de una valiosa herramienta: la teoría de congruencias, creada por Gauss. La manera moderna de introducir las congruencias es a través de las relaciones de equivalencia.

1.5.1 Relaciones de equivalencia

Definición 1.5.1 Sea X un conjunto. Una *relación en* X es un subconjunto R del producto cartesiano $X \times X$. Sean $x, y \in X$, decimos que x e y están *relacionados*, y lo denotamos por $x \sim y$ si $(x, y) \in R$.

Ejemplo 1.5.1 En $\mathbb{R}$, la relación de igualdad $x = y$ corresponde al subconjunto $R \subset \mathbb{R} \times \mathbb{R}$, $R = \{(a, a) : a \in \mathbb{R}\}$.

Definición 1.5.2 Sea X un conjunto. Una relación "$\sim$" en X se dice que es de *equivalencia* si dados $x, y, z \in X$ arbitrarios se satisfacen

(i) $x \sim x$ (*reflexividad*);

(ii) si $x \sim y$, entonces $y \sim x$ (*simetría*);

(iii) si $x \sim y$ y $y \sim z$, entonces $x \sim z$ (*transitividad*).

La relación de igualdad entre los elementos de un conjunto cualquiera es una relación de equivalencia.

Las relaciones de equivalencia son utilizadas para clasificar los elementos de un conjunto en subconjuntos con propiedades semejantes.

Definición 1.5.3 Sea $x \in X$, La *clase de equivalencia de* x es

$$\overline{x} = \{y \in X : y \sim x\}.$$

Un elemento y de $\overline{x}$ es llamado *representante de la clase.*

La idea del representante es la de un elemento que me permite recuperar toda la clase a partir de él. Más precisamente,

$$\text{si } x \in X \text{ y } y \in \overline{x}, \text{ entonces } \overline{x} = \overline{y}$$

(esta propiedad es fácil de probar a partir de la definición de relación de equivalencia). Luego tenemos la siguiente caracterización de las relaciones de equivalencia:

Proposición 1.5.1 Una relación de equivalencia particiona el espacio X. Esto es

(i) $X = \bigcup_{x \in X} \overline{x}$;

(ii) si $\overline{x} \neq \overline{y}$, entonces $\overline{x} \cap \overline{y} = \emptyset$.

El conjunto de las clases de equivalencia de "$\sim$" en X es llamado *espacio cociente* de X por "$\sim$", y se denota $\frac{X}{\sim}$.

Ejemplo 1.5.2 (Fracciones) Considere

$$X = \mathbb{Z} \times (\mathbb{Z} \setminus \{0\}).$$

Definimos en X la relación

$$(a, b) \equiv (c, d) \text{ si y sólo si } ad = bc.$$

Es fácil comprobar que "$\sim$" es una relación de equivalencia en X. Una fracción es una clase de equivalencia en ese conjunto; utilizamos la notación

$$\frac{a}{b} = \overline{(a, b)}.$$

Gracias a esto, para $c \neq 0$, ocurre que

$$\frac{ac}{bc} = \frac{a}{b},$$

propiedad bastante conocida de las fracciones.

1.5.2 Definición y propiedades de las congruencias

Ahora que conocemos el concepto de relación de equivalencia, podemos utilizar esta herramienta para estudiar las propiedades de divisibilidad de los números enteros. Definiremos una relación de equivalencia en $\mathbb{Z}$ que relacione números enteros con características de divisibilidad similares.

Definición 1.5.4 Dados enteros a, b, n con $n > 0$, decimos que a *es congruente con* b *módulo* n, y escribimos

$$a \equiv b (\bmod n)$$

si $n|(a-b)$. El número n es llamado *módulo de la congruencia.*
Si $n \not| (a-b)$ escribimos $a \not\equiv b (\bmod n)$ y decimos que a y b son *incongruentes módulo* n.

En particular,

$$a \equiv 0 (\bmod n) \text{ si y sólo si } n|a.$$

Además,

$$a \equiv b (\bmod n) \text{ si y sólo si } a \bmod n = b \bmod n,$$

esto es, el resto al dividir a por n y b por n es el mismo.

Proposición 1.5.2 La relación "$\equiv$" es de equivalencia en $\mathbb{Z}$.

El espacio cociente

$$\mathbb{Z}_n = \frac{\mathbb{Z}}{\equiv},$$

es llamado conjunto de los *enteros módulo* n. ¿Qué forma tiene $\mathbb{Z}_n$ y sus elementos (las clases)? La siguiente proposición responde a la pregunta.

Proposición 1.5.3 Si $a \in \mathbb{Z}$, entonces

$$\overline{a} = \{a + kn : k \in \mathbb{Z}\}.$$

Además

$$\mathbb{Z}_n = \{\overline{0}, \overline{1}, \ldots, \overline{n-1}\}$$

(*forma reducida* de las clases de $\mathbb{Z}_n$).

Definición 1.5.5 Un conjunto de n representantes, uno de cada una de las clases de restos $\overline{0}, \overline{1}, \ldots, \overline{n-1}$ o lo que es lo mismo, un conjunto de n enteros, incongruentes $\bmod n$, es llamado *sistema completo de restos* módulo n;

Dotaremos ahora a $\mathbb{Z}_n$ de una estructura natural, heredada de $\mathbb{Z}$. Definimos para $\overline{a}, \overline{b} \in \mathbb{Z}_n$, las operaciones *suma* y *producto* por

$$\overline{a} + \overline{b} = \overline{a+b}; \qquad \overline{a} \cdot \overline{b} = \overline{ab}. \tag{1.1}$$

Estas operaciones no dependen de los representantes elegidos. Si $\overline{a} = \overline{c}$ y $\overline{b} = \overline{d}$, entonces $n|(a-c)$, $n|(b-d)$ de donde

$$n|[(a+b) - (c+d)] = (a-c) + (b-d)$$

y

$$n|(ab - cd) = a(b - d) + d(a - c)$$

por linealidad de la división; por lo tanto

$$\overline{a} + \overline{b} = \overline{a + b} = \overline{c + d} = \overline{c} + \overline{d},$$
$$\overline{a} \cdot \overline{b} = \overline{ab} = \overline{cd} = \overline{c} \cdot \overline{d},$$

lo que prueba la buena definición de las operaciones en $\mathbb{Z}_n$. Estas operaciones heredan además las propiedades de las operaciones en $\mathbb{Z}$, las propiedades de *anillo* (ver la sección 1.7).

Definición 1.5.6 Sea $\overline{a} \in \mathbb{Z}_n$. Decimos que $\overline{a}$ es *inversible* si existe $\overline{b} \in \mathbb{Z}_n$ tal que $\overline{a} \cdot \overline{b} = \overline{1}$; este elemento $\overline{b}$ se denota por $\overline{a}^{-1}$ y es llamado *inverso de a módulo n*. El conjunto de los elementos inversibles de $\mathbb{Z}_n$ se denota por $\mathbb{Z}_n^*$.

1.5.3 La función φ de Euler y el Teorema Chino del Resto

Nos detenemos por el momento a estudiar la cardinalidad del conjunto $\mathbb{Z}_n^*$, mediante la función φ de Euler, además de un teorema relacionado, que nos será de gran utilidad, el Teorema Chino del Resto.

Definición 1.5.7 Definimos la *función φ de Euler* como la aplicación que asigna, a cada $n \in \mathbb{N}$, la cardinalidad de $\mathbb{Z}_n^*$.

La siguiente proposición nos da una caracterización de los elementos de $\mathbb{Z}_n^*$:

Proposición 1.5.4 Sea $\overline{a} \in \mathbb{Z}_n$. Entonces $\overline{a}$ es inversible si y sólo si $(a, n) = 1$.

Prueba. Tenemos que $(a, n) = 1$ si y sólo si existen enteros r, s tales que $ra + sn = 1$, esto es si y sólo si $ra \equiv 1 (\mathrm{mod}\, n)$ o equivalentemente $\overline{r} \cdot \overline{a} = \overline{1}$.

Como consecuencia de la última proposición,

$$\varphi(n) = \#\{1 \leq a \leq n : (a, n) = 1\}.$$

Definición 1.5.8 Un conjunto de $\varphi(n)$ representantes, uno de cada una de las clases de restos de $\mathbb{Z}_n^*$, o lo que es lo mismo, un conjunto de $\varphi(n)$ enteros, incongruentes $\mathrm{mod}\, n$ y coprimos con n, es llamado *sistema residual reducido módulo n*.

A continuación estudiaremos la forma que toma la función φ de Euler.

Proposición 1.5.5 Sea p primo, $\alpha \in \mathbb{N}$, entonces

$$\varphi(p^\alpha) = p^\alpha - p^{\alpha-1}.$$

Prueba. Supongamos que $1 \leq a \leq p^\alpha$, $(a, p^\alpha) \neq 1$. Entonces $p|a$, esto es, podemos escribir $a = pk$ con $k \in \mathbb{Z}$. Como $1 \leq pk \leq p^\alpha$, $1 \leq k \leq p^{\alpha-1}$ y luego $p^{\alpha-1}$ enteros entre 1 y p^α cumplen $(a, p^\alpha) \neq 1$.

Teorema 1.5.6 (Teorema chino del resto) Supongamos que $m_1, \ldots, m_t \in \mathbb{N}$ son primos dos a dos (esto es $(m_i, m_j) = 1$ si $i \neq j$). Si $\overline{a_i}$ son enteros, entonces existe un único elemento entero a módulo m tal que

$$a \equiv a_i (\bmod m_i), \quad i = 1, \ldots, t$$

donde $m = m_1 \cdots m_t$.

Prueba. Probemos la unicidad. Suponga que $b \equiv a_i (\bmod m_i)$ para todo i, entonces $a \equiv b (\bmod m_i)$ para todo i y de ahí $m_i | a - b$. Como los m_i son coprimos, entonces $m|a-b$, o equivalentemente, $a \equiv b (\bmod m)$. Ahora veamos la existencia. Sea $n_i = m/m_i$. Como $(m_i, n_i) = 1$, existen r_i, s_i enteros tales que $r_i m_i + s_i n_i = 1$. Sea

$$a = a_1 s_1 n_1 + a_2 s_2 n_2 + \cdots + a_t s_t n_t,$$

como $m_i | n_j$ para $i \neq j$, entonces

$$a \equiv a_i s_i n_i \equiv a_i (\bmod m_i).$$

Como consecuencia de este último teorema, la función

$$\Phi : \mathbb{Z}_m \to \mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2} \times \cdots \times \mathbb{Z}_{m_t}$$

definida por

$$\Phi(a (\bmod n)) = (a (\bmod m_1), a (\bmod m_2), \cdots, a (\bmod m_t))$$

es una biyección. Restringiendo a los elementos inversibles de $\mathbb{Z}_m$, vemos que

$$\Phi|_{Z_m^*} : \mathbb{Z}_m^* \to \mathbb{Z}_{m_1}^* \times \cdots \times \mathbb{Z}_{m_t}^*$$

es una biyección y por lo tanto, considerando la cardinalidad de los conjuntos involucrados:

$$\varphi(m) = \varphi(m_1)\varphi(m_2) \cdots \varphi(m_t).$$

Corolario 1.5.1 Si $m_1, \ldots, m_t$ son números naturales primos dos a dos, entonces

$$\varphi(m_1 \cdots m_t) = \varphi(m_1)\varphi(m_2) \cdots \varphi(m_t).$$

En particular, si $m \in \mathbb{N}$, podemos escribir m como $m = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_t^{\alpha_t}$, donde los p_i son primos distintos, y entonces

$$\varphi(m) = (p_1^{\alpha_1} - p_1^{\alpha_1 - 1})(p_2^{\alpha_2} - p_2^{\alpha_2 - 1}) \cdots (p_t^{\alpha_t} - p_t^{\alpha_t - 1}).$$

Teorema 1.5.7 (Euler) Sean n, a enteros con n positivo, $(a, n) = 1$. Entonces

$$a^{\varphi(n)} \equiv 1 (\bmod n).$$

Prueba. La función $f : \mathbb{Z}_n^* \to \mathbb{Z}_n^*$ definida por $f(\overline{b}) = \overline{a} \cdot \overline{b}$ es una biyección; de donde $f(\mathbb{Z}_n^*) = \mathbb{Z}_n^*$. Por lo tanto el producto de todos los elementos de $\mathbb{Z}_n^*$ es igual al producto de todas las imágenes mediante f,

$$\overline{b_1} \cdots \overline{b_{\varphi(n)}} = \overline{ab_1} \cdots \overline{ab_{\varphi(n)}} = \overline{a}^{\varphi(n)} \overline{b_1} \cdots \overline{b_{\varphi(n)}},$$

de donde $\overline{a}^{\varphi(n)} = \overline{1}$, o

$$a^{\varphi(n)} \equiv 1 (\bmod n).$$

Corolario 1.5.2 (Pequeño teorema de Fermat) Sea p primo, y a un entero tal que $p \nmid a$. Entonces

$$a^{p-1} \equiv 1 (\bmod p).$$

Prueba. Basta observar que $\varphi(p) = p - 1$ cuando p es primo.

1.5.4 Congruencias lineales

En esta sección estudiaremos la solución de las ecuaciones de la forma

$$ax \equiv b (\bmod n)$$

en la incógnita x, donde a, b, x y n son enteros, $n > 1$. Las ecuaciones del tipo anterior se denominan *congruencias lineales.*

Veamos el caso más simple:

Proposición 1.5.8 Supongamos que $(a, n) = 1$, entonces la congruencia

$$ax \equiv b (\bmod n)$$

posee una única solución módulo n.

Prueba. En efecto, como $(a, n) = 1$, entonces a es invertible módulo n y escribimos

$$x \equiv a^{-1} b (\bmod n),$$

donde a^{-1} es el inverso de a módulo n.

La generalización del resultado anterior es la siguiente:

Teorema 1.5.9 Supongamos que $(a, n) = d$. La congruencia

$$ax \equiv b(\bmod n) \tag{1.2}$$

posee solución si, y sólo si, $d|b$. En este caso, la congruencia tiene exactamente d soluciones módulo n.

Prueba. Si existe solución de (1.2), entonces existen x, k enteros tales que

$$ax - b = kn,$$

o $b = ax - kn$, y luego $d|b$. Recíprocamente, si $d|b$, podemos considerar la congruencia

$$\frac{a}{d}x \equiv \frac{b}{d}\left(\bmod \frac{n}{d}\right) \tag{1.3}$$

tiene una única solución t pues $\left(\frac{a}{d}, \frac{n}{d}\right) = 1$. Dicha solución es entonces solución de (1.2). Los elementos

$$t_j = t + j\frac{n}{d}, \quad 0 \leq j < d$$

son d soluciones de (1.3), incongruentes módulo n. Basta ahora ver que toda solución de (1.3) es congruente a algún t_j. En efecto, si $ax \equiv b(\bmod n)$, entonces

$$ax \equiv at(\bmod n),$$

de donde $a(x-t) \equiv 0(\bmod n)$ y luego $\frac{a}{d}(x-t) \equiv 0\left(\bmod \frac{n}{d}\right)$; ahora como $\frac{a}{d}$ es invertible módulo $\frac{n}{d}$,

$$x \equiv t\left(\bmod \frac{n}{d}\right).$$

Por lo tanto $x = t + k\frac{n}{d}$ donde $k \equiv j(\bmod d)$, para algún j, $0 \leq j < d$.

1.5.5 Aritmética modular

Veamos ahora el costo de realizar aritmética módulo n. Podemos ver los elementos de $\mathbb{Z}_n$ como $0, 1, \ldots, n-1$ con la suma y el producto

$$a \oplus b = (a + b) \bmod n,$$

$$a \odot b = (a \cdot b) \bmod n,$$

donde $m \bmod n$ es el resto de la división de m entre n. Como el número de dígitos de un número en $\mathbb{Z}_n$ es a lo más $\lfloor \log_2 n \rfloor + 1$, la suma en $\mathbb{Z}_n$ tiene complejidad $O(\log n)$ y el producto $O(\log^2 n)$, por lo que vimos anteriormente.

Requeriremos a menudo calcular $a^m \bmod n$; daremos para esto un algoritmo eficiente.

Algoritmo 1.5.1 (Exponenciación en $\mathbb{Z}_n$) Dados un elemento $a \in \mathbb{Z}_n$, y un número natural $m = (m_r \dots m_1 m_0)_2$, este algoritmo calcula la potencia m-ésima de a en $\mathbb{Z}_n$:

Datos:	a con $0 \leq a < n$ $m = (m_r \dots m_1 m_0)_2$
$e = 1$ $k = a$	
para $j = 0, \dots, r$	
	si $m_j = 1$
	$e = ek \bmod n$
	$k = k^2 \bmod n$
Salida:	$e = a^m \bmod n$

El costo del algoritmo es, considerando $r = O(\log m)$ iteraciones de costo $O(\log^2 n)$ cada una, igual a $O(\log^2 n)$ u

$$O(\log m \log^2 n).$$

Cálculo de inversos en $\mathbb{Z}_n$

Como sabemos, no todos los elementos de $\mathbb{Z}_n$ tienen inverso multiplicativo. De hecho,

$$a \in \mathbb{Z}_n \text{ es inversible si y sólo si } (a, n) = 1.$$

Si $(a, n) = 1$, existen enteros (no necesariamente únicos) r, s tales que

$$ra + sn = 1$$

y luego

$$ra \equiv 1 (\bmod n),$$

esto es, r es el inverso de a en $\mathbb{Z}_n$. Luego, para obtener r basta aplicar el algoritmo euclidiano extendido. De ahí vemos que el costo de calcular inversos en $\mathbb{Z}_n$ es $O(\log^3 n)$.

1.6 Grupos abelianos

1.6.1 Propiedades básicas

Definición 1.6.1 Un *grupo* es un par $(G, *)$, donde G es un conjunto no vacío y $* : G \times G \to G$ una aplicación (llamada *operación binaria*), tales que:

(i) $(a * b) * c = a * (b * c)$, para cualesquiera $a, b, c \in G$ (*asociatividad*);

(ii) existe un único elemento $e \in G$, llamado *identidad*, tal que $a * e = e * a = a$, para todo $a \in G$;

(iii) para cada $a \in G$ existe un único elemento $a^{-1} \in G$, llamado *inverso de* a, tal que $a * a^{-1} = a^{-1} * a = e$;

(iv) $a * b = b * a$, para todo $a, b \in G$ (*conmutatividad*).

Nota. La verdadera definición de grupo corresponde a las propiedades (i), (ii) y (iii) solamente. Al agregar la propiedad (iv) tenemos lo que se denomina un *grupo abeliano*. Sin embargo en adelante sólo trabajaremos con grupos abelianos, por lo que los llamaremos simplemente grupos.

Si no hay peligro de confusión simplemente denotaremos al grupo $(G, *)$ por G. Además, comúnmente escribiremos ab en vez de $a * b$; y la notación a^n con $a \in G$ y $n \in \mathbb{Z}$ denota

$$a^n = \begin{cases} \underbrace{a * a * \cdots * a}_{n \text{ términos}}, & \text{si } n > 0; \\ e, & \text{si } n = 0; \\ \underbrace{a * a * \cdots * a}_{(-n) \text{ términos}}, & \text{si } n < 0. \end{cases}$$

Definición 1.6.2 Un grupo G es *finito* si G es un conjunto finito; en este caso se denota por $|G|$ al número de elementos de G, el cual se denomina *orden* de G.

Una notación que se utiliza en algunos grupos es la *notación aditiva*. La operación de grupo se denota por el símbolo "+" (y se denomina *suma*); el elemento neutro de la operación se denota 0 y el inverso de un elemento $a \in G$ se denota $-a$. Además se escribe

$$a - b = a + (-b),$$

y a^n se denota simplemente na, para $n \in \mathbb{Z}$.

Ejemplos 1.6.1 Los conjuntos $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$ y $\mathbb{Z}_n$ (para $n \in \mathbb{N}$) definidos anteriormente son grupos con la operación "+", y utilizan la notación aditiva. El conjunto $\mathbb{Z}_n^*$ es un grupo con la operación "$\cdot$".

Definición 1.6.3 Un subconjunto no vacío H de G, es llamado *subgrupo* de G, si H es un grupo con la operación restringida de G, esto es, H es un grupo con $*|H \times H$.

La siguiente caracterización de los subgrupos es muy utilizada en la práctica:

Proposición 1.6.1 Sea $H \subset G$. Entonces H es un subgrupo de G si y sólo si $H \neq \emptyset$ y

$$x * y^{-1} \in H, \text{ para cualesquiera } x, y \in H.$$

Prueba. Si H es un subgrupo de G, es evidente que dado $x, y \in H$, entonces $y^{-1} \in H$ y por lo tanto $x * y^{-1} \in H$. Recíprocamente, sea $H \subset G$, $H \neq \emptyset$ tal que $x * y^{-1} \in H$ para $x, y \in H$. Como $H \neq \emptyset$ entonces existe $x \in H$ y luego $e \in H$ pues $e = x * x^{-1}$. Dado $y \in H$, entonces $y^{-1} \in H$ pues $y^{-1} = e * y^{-1}$ (y $e \in H$). Además para $x, y \in H$ cualesquiera, $x * y = x * (y^{-1})^{-1} \in H$. De este modo H con la operación restringida de G es un grupo.

Proposición 1.6.2 Sea $\{H_i\}_{i \in I}$ una familia arbitraria de subgrupos de un grupo G. Entonces $\bigcap_{i \in I} H_i$ es un subgrupo de G.

Prueba. Como $e \in G$ pertenece a H_i para todo $i \in I$, entonces $e \in \bigcap_{i \in I} H_i = H$ de donde este conjunto es no vacío. Dados $x, y \in H$, tenemos que $x, y \in H_i$ para $i \in I$ cualquiera. Como los H_i son subgrupos, $xy^{-1} \in H_i$ para todo $i \in I$; pero de ahí, $xy^{-1} \in H$. Por lo tanto H es un subgrupo.

Sea $S \subset G$ un conjunto cualquiera y sea

$$\mathscr{F} = \{H \text{ subgrupo de } G : S \subset H\}.$$

Como $G \in \mathscr{F}$, entonces $\mathscr{F} \neq \emptyset$. Por la proposición anterior, $\bigcap_{H \in \mathscr{F}} H$ es también un subgrupo de G. Denotamos $\langle S \rangle = \bigcap_{H \in \mathscr{F}} H$, es fácil ver que todo subgrupo que contiene a S contiene forzosamente al subgrupo $\langle S \rangle$, lo que nos lleva a la siguiente definición.

Definición 1.6.4 Sea $S \subset G$. El *subgrupo generado por* S, denotado $\langle S \rangle$, es el menor subgrupo de G que contiene a S.

De aquí es fácil ver que si $S = \emptyset$, entonces $\langle S \rangle = \{e\}$, el conjunto formado por el elemento neutro de G. En el caso no trivial, tenemos:

Proposición 1.6.3 Sea $S \subset G$ no vacío. Si $S^{-1} = \{s^{-1} : s \in S\}$, entonces

$$\begin{aligned} \langle S \rangle &= \{t_1 \cdots t_n : t_i \in S \cup S^{-1}, n \in \mathbb{N}\}. \\ &= \{s_1^{\alpha_1} \cdots s_r^{\alpha_r} : s_i \in S, \alpha_j \in \mathbb{Z}, r \in \mathbb{N}\} \end{aligned}$$

Prueba. Sea

$$H = \{t_1 \cdots t_n : t_i \in S \cup S^{-1},\ n \in \mathbb{N}\}.$$

Es obvio que $S \subset H$ (corresponde a $n = 1$ en la definición de H). Veamos que H es un subgrupo de G: sean $x, y \in H$,

$$x = t_1 \cdots t_r,\ t_i \in S \cup S^{-1}, \qquad y = u_1 \cdots u_{r'},\ u_i \in S \cup S^{-1},$$

entonces

$$xy^{-1} = t_1 \cdots t_r (u_1)^{-1} \cdots (u_i)^{-1}$$

con $t_i, u_j^{-1} \in S \cup S^{-1}$ para cualesquiera i, j, de donde $xy^{-1} \in H$. Sea ahora K un subgrupo cualquiera de G tal que $S \subset K$. Entonces es obvio que $S^{-1} \subset K$, y cualquier producto finito $t_1 \cdots t_r$ de elementos $t_i \in S \cup S^{-1}$ está en K. Por lo tanto $H \subset K$. Siendo H el menor subgrupo que contiene a S se sigue que $H = \langle S \rangle$.

En el caso particular en el que $S = \{a\}$, $a \in G$ denotamos simplemente $\langle S \rangle = \langle a \rangle$. En este caso,

$$\langle a \rangle = \{a^n : n \in \mathbb{Z}\}$$

es llamado el *subgrupo cíclico generado por a*. El *orden de a*, denotado $o(a)$, es el orden del subgrupo generado por a: $o(a) = |\langle a \rangle|$.

En caso que $G = \langle a \rangle$, entonces G se denomina un *grupo cíclico*.

Proposición 1.6.4 Todo subgrupo de un grupo cíclico, es también cíclico.

Prueba. Sea $G = \langle a \rangle$ y H un subgrupo de G; podemos suponer que $H \neq \{e\}$. Sea

$$k = \min\{n \in \mathbb{N} :\ a^n \in H\},$$

observe que el conjunto anterior es no vacío pues si $a^{-n} \in H$, $n > 0$ entonces $a^n = (a^{-n})^{-1} \in H$. Además, k existe por el principio de la buena ordenación (no vamos a resaltar esto de ahora en adelante).

Afirmamos que $H = \langle a^k \rangle$: en efecto, sea $a^n \in H$ cualquiera, por el algoritmo de la división en $\mathbb{Z}$, existen enteros $q, r \in \mathbb{Z}$ tales que $n = qk + r$, $0 \leq r < k$. Entonces

$$a^r = a^n \cdot (a^k)^{-q} \in H,$$

de donde $r = 0$, pues de otro modo se contradice la minimalidad de k; y por lo tanto $a^n = (a^k)^q \in \langle a^k \rangle$.

Sea ahora H un subgrupo de G. Definimos en G la relación $x \sim y$ por

$$x \sim y \text{ si y sólo si } x * y^{-1} \in H,$$

esto es, si existe $z \in H$ tal que $x = yz$.

Proposición 1.6.5 La relación "$\sim$" es una relación de equivalencia en G. Además, para $x \in G$

$$\overline{x} = \{y \in G : x \sim y\} = Hx = \{hx : h \in H\}.$$

Así queda G particionado en clases de equivalencia, y podemos escribir

$$G = \bigcup_{i \in I}^{\cdot} Hx_i \text{ (unión disjunta).}$$

El espacio cociente obtenido se denota por G/H. Ahora veamos que $\#(Hx) = \#H$: en efecto, la aplicación $f : H \to Hx$ definida por $f(h) = hx$ es una biyección.

Como G es finito, entonces I es finito. Sea $n = \#I$. Así podemos escribir

$$G = \bigcup_{i=1}^{n\cdot} Hx_i,$$

y por lo tanto

$$\#G = \sum_{i=1}^{n} \#(Hx_i) = \sum_{i=1}^{n} \#H = n \cdot \#H.$$

Esto se resume en el siguiente teorema.

Teorema 1.6.6 (Lagrange) Si G es finito y H es un subgrupo de G, entonces $|H|$ divide a $|G|$.

Corolario 1.6.1 Sea G un grupo finito. Si $|G| = p$, p es primo, entonces G es cíclico.

Prueba. Sea $a \in G$, $a \neq e$ y sea $H = \langle a \rangle$. Por el teorema de Lagrange, $|H|$ divide a p. Como p es primo, $|H|$ es 1 o p; pero $|H| \neq 1$. Por lo tanto $|H| = p = |G|$ y $G = H$.

Dado un elemento a en un grupo G, consideramos el subgrupo cíclico generado por a, $\langle a \rangle$. Si este conjunto es finito, entonces la sucesión $a^0, a^1, a^2, \ldots$ no puede tener infinitos elementos distintos. Luego existirán i, j con $i < j$ tales que $a^i = a^j$. Tomando $m = j - i > 0$ tenemos que $a^m = e$. Sea ahora

$$k = \min\{m > 0 : a^m = e\}.$$

Afirmamos que $\langle a \rangle = \{e, a, a^2, \cdots, a^{k-1}\}$. En efecto, sea $n \in \mathbb{Z}$ cualquiera, luego por el algoritmo de la división

$$n = qk + r \text{ donde } 0 \leq r < k,$$

y

$$a^n = (a^k)^q \cdot a^r = e^q \cdot a^r = a^r.$$

Los elementos a^j para $0 \leq j < k$ son distintos, pues si $a^i = a^j$ para $0 \leq i < j < k$, entonces $a^{j-i} = e$ con $0 < j - i < k$, absurdo.

Además, si $a^n = e$ en lo anterior, entonces $a^r = e$ de donde forzosamente $r = 0$; y $k|n$.

Así tenemos la siguiente proposición:

Proposición 1.6.7 Sea $a \in G$. Si $\langle a \rangle$ es finito, entonces

$$o(a) = \min\{k \in \mathbb{N} : a^k = e\},$$

y

$$\langle a \rangle = \{a, a^2, \ldots, a^{o(a)}\}.$$

Además, si $a^l = e$, entonces $o(a)|l$.

Corolario 1.6.2 Si G es finito y $a \in G$, entonces

$$a^{|G|} = e.$$

Prueba. Por el teorema de Lagrange, $o(a)$ divide al orden de G, esto es $|G| = k \cdot o(a)$ con $k \in \mathbb{Z}$, y por lo tanto

$$a^{|G|} = (a^{o(a)})^k = e^k = e.$$

Una consecuencia trivial de este teorema es el teorema de Euler 1.5.7. Basta aplicar el corolario anterior a $G = \mathbb{Z}_n^*$, $\overline{a} \in \mathbb{Z}_n^*$, ya que $|G| = \varphi(n)$.

Ahora daremos un criterio para determinar si un grupo es cíclico. Dado un grupo finito G, sea

$$\exp G = \min\{m \in \mathbb{N} : a^m = e, \text{ para todo } a \in G\}.$$

Necesitaremos alguna preparación previa:

Lema 1.6.1 Sean g, h elementos de un grupo G tales que $(o(g), o(h)) = 1$. Entonces

$$o(gh) = o(g)o(h).$$

Prueba. En efecto, supongamos que $(gh)^r = e$. Entonces

$$k = g^r = h^{-r} \in \langle g \rangle \cap \langle h \rangle.$$

Luego $o(k)|o(g)$ y $o(k)|o(h)$ y por lo tanto $o(k) = 1$. De ahí $(gh)^r = e$ implica $g^r = e = h^r$. Luego $o(g)|r$ y $o(h)|r$ y por lo tanto $o(g)o(h)|r$. Esto prueba que $o(g)o(h)|o(gh)$. Por otro lado

$$(gh)^{o(g)o(h)} = (g^{o(g)})^{o(h)}(h^{o(h)})^{o(g)} = e,$$

de donde $o(gh)|o(g)o(h)$. Por lo tanto $o(gh) = o(g)o(h)$.

Lema 1.6.2 Sea G un grupo finito, $g \in G$ un elemento de orden maximal. Entonces $\exp G = o(g)$.

Prueba. Debemos mostrar que $h^{o(g)} = e$ para todo $h \in G$, lo cual es suficiente para la conclusión del teorema. Escribimos $o(g) = p_1^{e_1} \cdots p_s^{e_s}$, $o(h) = p_1^{f_1} \cdots p_s^{f_s}$ donde los p_i son primos distintos y $e_i, f_i \geq 0$. Si $h^{o(g)} \neq e$, entonces algún $f_i > e_i$; podemos asumir sin problema que $f_1 > e_1$. Hacemos $g' = g^{p_1^{e_1}}$, $h' = h^{p_2^{f_2} \cdots p_s^{f_s}}$. Entonces $o(g') = p_2^{e_2} \cdots p_s^{e_s}$ y $o(h) = p_1^{f_1}$. Por el lema anterior, $o(g'h') = p_1^{f_1} p_2^{e_2} \cdots p_s^{e_s} > o(g)$, contradiciendo la maximalidad de $o(g)$.

Teorema 1.6.8 Un grupo finito G es cíclico si y sólo si $|G| = \exp G$.

Prueba. Si G es cíclico, entonces existe $a \in G$ con

$$|G| = o(a) \leq \exp G.$$

Por otro lado, supongamos que $|G| = \exp G$. Escogiendo un elemento g de orden maximal, por el lema anterior tenemos que

$$|G| = \exp G = o(g) = |\langle g \rangle|$$

y por lo tanto $G = \langle g \rangle$.

1.6.2 Grupos cocientes

Retornamos ahora al estudio del espacio cociente G/H. Recordemos que

$$\frac{G}{H} = \{aH : a \in G\}.$$

Deseamos dotar a G/H de una estructura natural de grupo. Definimos la operación "$*$" en G/H por

$$(aH) * (bH) = (a * b)H = (ab)H.$$

La operación dada está bien definida: si $aH = xH$, $bH = yH$ para $x, y \in G$, entonces $ax^{-1} \in H$, $by^{-1} \in H$ y

$$(ab)(xy)^{-1} = (ax^{-1})(by^{-1}) \in H,$$

de donde $(ab)H = (xy)H$.

Las propiedades de grupo se verifican trivialmente en G/H: el elemento neutro de G/H es $eH = e$ y el inverso de un elemento aH es $a^{-1}H$.

Proposición 1.6.9 Sea H un subgrupo de G. Entonces G/H con la operación

$$(aH) * (bH) = (ab)H$$

es un grupo, llamado *grupo cociente* de G sobre H.

Como consecuencia de la prueba del teorema de Lagrange, tenemos de inmediato:

Proposición 1.6.10 Si G es finito y H es un subgrupo de G, entonces

$$|G/H| = \frac{|G|}{|H|}.$$

Prueba Recuerde que en la prueba del teorema, $|G| = n|H|$, donde n es el número de clases de equivalencia de G por H, esto es $n = |G/H|$.

Observación. En la notación aditiva una clase aH se escribe como $a + H$, y así

$$\frac{G}{N} = \{a + H : a \in G\}.$$

Ejemplo 1.6.2 Sea $G = \mathbb{Z}$, $n \in \mathbb{N}$, $H = \langle n \rangle = \{kn : n \in \mathbb{Z}\}$. Entonces $x, y \in \mathbb{Z}$ están relacionados, $x \sim y$ si y sólo si $x - y \in H$, esto es, si y sólo si $n|x - y$. Concluimos que

$$\frac{\mathbb{Z}}{\langle n \rangle} = \mathbb{Z}_n.$$

Por lo tanto $\mathbb{Z}_n$ es un grupo con respecto a la suma. Observe que no hemos dicho nada con respecto al producto de $\mathbb{Z}_n$; esa discusión se hará más adelante.

1.6.3 Homomorfismos

Definición 1.6.5 Sean $(G, *)$, $(H, \circ)$ grupos. Decimos que la aplicación $f : G \to H$ es un *homomorfismo* si preserva la estructura de grupo, esto es

$$f(a * b) = f(a) \circ f(b),$$

para cualesquiera $a, b \in G$.

Proposición 1.6.11 Sea $f : (G, *) \to (H, \circ)$ un homomorfismo de grupos.

(i) $f(e_G) = e_H$;

(ii) $f(g^{-1}) = f(g)^{-1}$ para todo $g \in G$;

(iii) si T es un subgrupo de G, entonces $f(T)$ es un subgrupo de H;

(iv) si Y es un subgrupo de H, entonces $f^{-1}(Y)$ es un subgrupo de G;

Prueba.

(i) $f(e_G)f(e_G) = f(e_Ge_G) = f(e_G) = f(e_G)e_H$, de donde $f(e_G) = e_H$.

(ii) Es claro pues $f(g)f(g^{-1}) = f(gg^{-1}) = f(e_G) = e_H$.

(iii) $f(T) \neq \emptyset$ pues $e_H = f(e_G) \in f(T)$ ($e_G \in T$). Si $x, y \in f(T)$, $x = f(x')$, $y = f(y')$ con $x', y' \in T$, entonces

$$xy^{-1} = f(x')f(y')^{-1} = f(x')f(y'^{-1}) = f(x'y'^{-1}) \in f(T)$$

dado que $x'y'^{-1} \in T$.

(iv) $f^{-1}(Y) \neq \emptyset$ dado que $f(e_G) = e_H \in Y$ implica $e_G \in f^{-1}(Y)$. Si $x, y \in f^{-1}(Y)$ ($f(x) \in Y$, $f(y) \in Y$) entonces

$$f(xy^{-1}) = f(x)f(y)^{-1} \in Y$$

y por lo tanto $xy^{-1} \in f^{-1}(Y)$.

Definición 1.6.6 Sea $f : (G, *) \to (H, \circ)$ un homomorfismo de grupos.

(i) f es un *monomorfismo* si f es inyectiva,

(ii) f es un *epimorfismo* si f es sobreyectiva.

Definición 1.6.7 Sea $f : (G, *) \to (H, \circ)$ un homomorfismo. Decimos que f es un *isomorfismo* si f es biyectiva, esto es si y sólo si f es monomorfismo y epimorfismo. En este caso, decimos que los grupos G y H son *isomorfos*, y denotamos $G \approx H$.

La existencia de un isomorfismo entre dos grupos nos dice que estos son casi "idénticos", ya que poseen la misma estructura. Por ello no extraña que en ocasiones, en vez de escribir $G \approx H$ se escriba $G = H$, como si se tratara del mismo espacio. Es uno de los conceptos más importantes del álgebra. La relación "$\approx$" es una relación de equivalencia en la clase de todos los grupos.

Definición 1.6.8 El conjunto

$$\text{Ker}(f) = f^{-1}(e_H) = \{g \in G : f(g) = e_H\}$$

es llamado *núcleo de* f; y es un subgrupo de G.

Observe además que la *imagen de* f, a la que denotaremos por $\mathrm{Im}(f) = f(G)$ es un subgrupo de H.

Proposición 1.6.12 Un homomorfismo $f : G \to H$ es un monomorfismo si y sólo si $\mathrm{Ker}(f) = \{e_G\}$.

Prueba. Si f es un monomorfismo, dado $y \in \mathrm{Ker}(f)$,

$$f(y) = e_H = f(e_G),$$

de donde $y = e_G$. Por lo tanto $\mathrm{Ker}(f) = \{e_G\}$. Por otro lado, supongamos que $\mathrm{Ker}(f) = \{e_G\}$; sean $x, y \in G$ tales que $f(x) = f(y)$. Entonces

$$f(xy^{-1}) = f(x)f(y)^{-1} = e_H$$

de donde $xy^{-1} \in \mathrm{Ker}(f) = \{e_G\}$, esto es $x = y$.

Teorema 1.6.13 (Teorema fundamental de los homomorfismos) Sea $f : (G, *) \to (H, \circ)$ un homomorfismo de grupos. Entonces

$$\frac{G}{\mathrm{Ker}(f)} \approx \mathrm{Im}(f).$$

Prueba. Definimos el homomorfismo $\tilde{f} : G/\mathrm{Ker}(f) \to H$ por

$$\tilde{f}(a\mathrm{Ker}(f)) = f(a).$$

La aplicación está bien definida: si $a\mathrm{Ker}(f) = b\mathrm{Ker}(f)$, entonces $ab^{-1} \in \mathrm{Ker}(f)$ y

$$f(a)f(b)^{-1} = f(ab^{-1}) = e_H,$$

esto es, $f(a) = f(b)$. Además $\tilde{f}$ es inyectiva: si $a\mathrm{Ker}(f) \in \mathrm{Ker}(\tilde{f})$, entonces

$$f(a) = \tilde{f}(a\mathrm{Ker}(f)) = e_H,$$

de donde $a \in \mathrm{Ker}(f)$ y $a\mathrm{Ker}(f) = \mathrm{Ker}(f) = e_G\mathrm{Ker}(f)$. Por lo tanto $\tilde{f}$ es un isomorfismo sobre su imagen $\mathrm{Im}(\tilde{f}) = \mathrm{Im}(f)$.

Corolario 1.6.3 Si $f : (G, *) \to (H, \circ)$ es un epimorfismo, entonces

$$\frac{G}{\mathrm{Ker}(f)} \approx H.$$

1.7 Anillos e Ideales

Definición 1.7.1 Un *anillo* es una terna $(A, +, \cdot)$, donde A es un conjunto no vacío, y dos operaciones $+, \cdot : A \times A$, llamadas *suma* y *producto*, respectivamente, tales que: dados $a, b, c \in A$,

(i) $(A, +)$ es un grupo, donde el elemento neutro para la suma se denota 0 y el inverso de un elemento a se denota $-a$ (notación aditiva);

(ii) $a \cdot (b \cdot c) = (a \cdot b) \cdot c$ (*asociatividad* del producto);

(iii) $a \cdot b = b \cdot a$ (*conmutatividad* del producto);

(iv) existe un elemento $1 \in A$ tal que $a \cdot 1 = 1 \cdot a = a$ (*elemento neutro* para el producto);

(v) se satisface la *ley distributiva*

$$(a + b) \cdot c = a \cdot c + b \cdot c.$$

Nota. Las propiedades (iii) y (iv) no corresponden a la verdadera definición de anillo, son las propiedades (i), (ii), (v), además de la ley distributiva $c \cdot (a + b) = c \cdot a + c \cdot b$ las que corresponden a la definición. La definición que dimos es la de *anillo conmutativo con unidad*. Como a lo largo del trabajo sólo trabajaremos con este tipo de anillos, los llamaremos simplemente anillos.

Si no hay peligro de confusión denotaremos A en vez de $(A, +, \cdot)$. Se acostumbra omitir el símbolo del producto, escribiendo $ab = a \cdot b$, mientras se mantiene el de la suma. Dado $a \in A$ denotamos

$$a^n = \underbrace{a \cdot a \cdot \cdots \cdot a}_{n \text{ términos}}.$$

Definición 1.7.2 Si para $a \in A$, existe $b \in A$ tal que $ba = ab = 1$ se dice que a es *inversible* y se denota $b = a^{-1}$. El conjunto de los elementos inversibles de A se denota A^*.

Definición 1.7.3 Un anillo A se dice que es un *dominio de integridad* si

$$ab = 0 \text{ con } a, b \in A \text{ implica que } a = 0 \text{ o } b = 0.$$

Definición 1.7.4 Un anillo A se dice que es un *campo* si y sólo si $(A \setminus \{0\}, \cdot)$ es un grupo, o lo que es lo mismo, si $A^* = A \setminus \{0\}$ (todos los elementos no nulos de A son inversibles).

Proposición 1.7.1 Todo campo es un dominio de integridad.

Prueba. En efecto, supongamos que $a \cdot b = 0$ con $a \neq 0$; entonces existe $a^{-1} \in A$ y

$$b = 1 \cdot b = (a^{-1} \cdot a) \cdot b = a^{-1} \cdot (a \cdot b) = a^{-1} \cdot 0 = 0.$$

Ejemplos 1.7.1 Los conjuntos $\mathbb{Q}$, $\mathbb{R}$ son ejemplos claros de campos; el conjunto $\mathbb{Z}$ es un ejemplo de dominio de integridad. El conjunto $\mathbb{Z}_n$ es un anillo: si n es compuesto, entonces $\mathbb{Z}_n$ no es un dominio de integridad, pues escribiendo $n = ab$ con $1 < a, b < n$, tenemos

$$\overline{a} \cdot \overline{b} = \overline{0}$$

con $\overline{a} \neq \overline{0}$, $\overline{b} \neq \overline{0}$; en cambio si n es primo,

$$\mathbb{Z}_n^* = \{\overline{k} \in \mathbb{Z}_n : (k, n) = 1\} = \{\overline{k} \in \mathbb{Z}_n : \overline{k} \neq \overline{0}\} = \mathbb{Z}_n \setminus \{\overline{0}\}$$

y $\mathbb{Z}_n$ es un campo.

Notemos ahora que en $\mathbb{Z}_n$,

$$n \cdot \overline{1} = \underbrace{\overline{1} + \overline{1} + \cdots + \overline{1}}_{n \text{ veces}} = \overline{n} = \overline{0}$$

mientras que en $\mathbb{Z}$,

$$n \cdot 1 = n \neq 0, \qquad \text{para todo } n \in \mathbb{N}.$$

Esto nos lleva a la siguiente definición:

Definición 1.7.5 Sea A un dominio de integridad, $1 \in A$. Decimos que la *característica de* A es 0 si

$$n \cdot 1 = \underbrace{1 + 1 + \cdots + 1}_{n \text{ veces}} \neq 0, \text{ para todo } n \in \mathbb{N};$$

en caso contrario, esto es si existe $n \in \mathbb{N}$ con $n \cdot 1 = 0$, definimos la *característica de* A como el menor de tales n. Denotamos por $\text{car}(A)$ a la característica de A.

Proposición 1.7.2 Sea A un dominio de integridad. Si $\text{car}(A) \neq 0$, entonces es un primo.

Prueba. Sea $\text{car}(A) = n$, $n > 0$ compuesto. Escribimos $n = ab$ con $1 < a, b < n$, de donde $a \cdot 1 \neq 0$, $b \cdot 1 \neq 0$ por definición. Pero

$$n \cdot 1 = (a \cdot 1) \cdot (b \cdot 1) = 0,$$

lo cual es absurdo pues A es un dominio de integridad.

Definición 1.7.6 Un subconjunto no vacío B de A se dirá un *subanillo* de A si y sólo si es un anillo con las operaciones inducidas de A, salvo posiblemente por la condición (iv) de la definición de anillo (la existencia del elemento neutro para el producto, que puede bien darse o no).

Ejemplos 1.7.2 $\mathbb{Z}$ es un subanillo de $\mathbb{Q}$; $\mathbb{Q}$ es subanillo de $\mathbb{R}$.

Al igual que los subgrupos, los subanillos poseen una caracterización:

Proposición 1.7.3 Un subconjunto no vacío B es un subanillo de A si y sólo si para cualesquiera $a, b \in B$:

(i) $a \cdot b \in B$;

(ii) $a - b \in B$.

La prueba de esta proposición es análoga a la de la proposición 1.6.1.

Definición 1.7.7 Un subconjunto no vacío J de A se dirá un *ideal* si para cualesquiera $a, b \in J$, $r \in A$:

(i) $a - b \in J$;

(ii) $ra \in J$.

Ejemplo 1.7.3 El conjunto $2\mathbb{Z} = \{2n : n \in \mathbb{Z}\}$ es un ideal de $\mathbb{Z}$.

La prueba de las dos siguientes afirmaciones se sigue de la definición de ideal:

Definición 1.7.8 Sea $a \in A$. El *ideal generado por* a es el conjunto

$$(a) = \{\lambda a : \lambda \in A\}$$

(el cual es, efectivamente, un ideal, el menor que contiene a a).

Proposición 1.7.4 Sean I, J ideales de A, el conjunto

$$I + J = \{x + y : x \in I, y \in Y\}$$

es un ideal.

Definición 1.7.9 Un ideal $M \subsetneq A$ se dirá *maximal* si $M \subset Y \subset A$ con Y ideal implica que $M = Y$ o $Y = A$.

Definición 1.7.10 Un ideal $P \subsetneq A$ se dirá *primo* si dados $a, b \in A$ con $ab \in P$, se tiene que $a \in P$ o $b \in P$.

Proposición 1.7.5 Todo ideal maximal es primo.

Prueba. Sea M un ideal maximal de A, y sean $a, b \in A$ tales que $ab \in M$; debemos probar que $a \in M$ o $b \in M$. Por reducción al absurdo, supongamos que $a \notin M$ y $b \notin M$. Entonces $M + (a)$ y $M + (b)$ son ideales con

$$M \subsetneq M + (a), \qquad M \subsetneq M + (b).$$

Como M es maximal, entonces $M + (a) = A = M + (b)$, de donde existen $m_1, m_2 \in M$ y $\lambda, \gamma \in A$ tales que

$$m_1 + \lambda a = 1 = m_2 + \gamma b,$$

y por lo tanto

$$1 = (m_1 + \lambda a)(m_2 + \lambda a) = m_1 m_2 + m_1 \gamma b + m_2 \lambda a + \lambda \gamma ab \in M,$$

ya que cada uno de los sumandos al lado derecho de la ecuación pertenecen a M. Luego $1 \in M$, lo cual implica que $M = A$, absurdo.

Sea I un ideal de A. Como $(I, +)$ es un subgrupo de $(A, +)$ podemos definir el grupo cociente $(A/I, +)$ de modo que

$$\frac{A}{I} = \{a + I : a \in A\},$$

y

$$(a + I) + (b + I) = (a + b) + I.$$

Queremos dotar a A/I de una estructura de anillo. Definamos $\cdot : A/I \times A/I \to A/I$ por

$$(a + I) \cdot (b + I) = ab + I.$$

Veamos que "$\cdot$" está bien definida: en efecto, si $a + I = c + I$, $b + I = d + I$ entonces $a - c \in I$, $b - d \in I$ y

$$ab - cd = b(a - c) - (-c)(b - d) \in I,$$

de donde $ab + I = cd + I$. Es fácil comprobar que $(A/I, +, \cdot)$ es un anillo. Todo esto se resume en el siguiente teorema.

Teorema 1.7.6 Si I es un ideal de A, entonces existe el *anillo cociente* $(A/I, +, \cdot)$ con las operaciones

$$(a + I) + (b + I) = (a + b) + I, \qquad (a + I) \cdot (b + I) = a \cdot b + I.$$

Ejemplo 1.7.4 Sea $n \in \mathbb{N}$, entonces

$$(n) = \{\lambda n : \lambda \in \mathbb{Z}\},$$

y

$$\frac{\mathbb{Z}}{(n)} = \mathbb{Z}_n,$$

el conjunto de los enteros módulo n.

Proposición 1.7.7 Un anillo A es un campo si y sólo si sus únicos ideales son $\{0\}$ y A.

Prueba. Supongamos que A es un campo, e I un ideal de A. O bien $I = \{0\}$, o existe $a \in I$, $a \neq 0$. Entonces $1 = a^{-1}a \in I$, de donde $I = A$ (ya que todo elemento $b \in A$ cumple $b = b \cdot 1 \in I$ si $1 \in I$, de donde $I \subset A$).

Recíprocamente, si A es un anillo cuyos únicos ideales son $\{0\}$ y A, entonces, dado $b \in A$, $b \neq 0$, el ideal generado por b es forzosamente igual a A, $(b) = A$. Por lo tanto existe $c \in A$ tal que $cb = 1$. Como todo elemento de A es inversible, A es un campo.

Definición 1.7.11 Sean $(A, +_A, \cdot_A), (B, +_B, \cdot_B)$ anillos. Sea $f : A \to B$ una aplicación. Decimos que f es un *homomorfismo de anillos* si para cualesquiera $a, b \in A$:

(i) $f(a +_A b) = f(a) +_B f(b)$,

(ii) $f(a \cdot_A b) = f(a) \cdot_B f(b)$.

El conjunto

$$\mathrm{Ker}(f) = \{a \in A : f(a) = 0\}$$

es llamado el *núcleo de* f.

Definición 1.7.12 Sea $f : A \to B$ un homomorfismo de anillos. Decimos que

(i) f es un *monomorfismo* si f es inyectiva;

(ii) f es un *epimorfismo* si f es sobreyectiva;

(iii) f es un *isomorfismo* si f es biyectiva, esto es, si y sólo si es monomorfismo y epimorfismo.

Ejemplo 1.7.5 Sea I un ideal del anillo A. La aplicación $\pi : A \to A/I$ definida por

$$\pi(a) = a + I,$$

es un epimorfismo, llamado *proyección canónica.*

La prueba de la siguiente proposición es similar a la de la proposición 1.6.11 y no la haremos.

Lema 1.7.1 Sea $f : A \to B$ un homomorfismo de anillos. Entonces

(i) si I es un ideal de B, entonces $f^{-1}(I)$ es un ideal de A,

(ii) si J es un ideal de A y f es sobreyectiva, entonces $f(J)$ es un ideal de B.

Teorema 1.7.8 Sea A un anillo, P, M ideales de A.

(i) P es primo en A si y sólo si A/P es un dominio.

(ii) M es maximal en A si y sólo si A/M es un campo.

Prueba.

(i) Supongamos que P es primo en A. Sean $a + P$, $b + P \in A/P$ tales que

$$(a + P)(b + P) = 0 + P = P,$$

entonces $ab + P = P$ y $ab \in P$, de donde $a \in P$ o $b \in P$, esto es $a + P = P$ o $b + P = P$. Recíprocamente, si A/P es un dominio entonces para $a, b \in A$ con $ab \in P$, tenemos

$$(a + P)(b + P) = ab + P = P,$$

de donde $a + P = P$ o $b + P = P$, esto es $a \in P$ o $b \in P$.

(ii) Utilizaremos el criterio dado por el teorema 1.7.7. Consideremos la proyección canónica $\pi : A \to A/M$, $\pi(a) = a + M$. Supongamos que M es maximal. Sea I un ideal de A/M, entonces $\pi^{-1}(I)$ es un ideal de A que contiene a M. Como M es maximal, entonces $M = \pi^{-1}(I)$ o $\pi^{-1}(I) = A$. En el primer caso, dado $a + M \in I$, si $a \in A$, entonces $\pi(x) = a + M$, entonces $a \in M$. Por lo tanto $a + M = M$ y tenemos $I = \{M\}$. En el segundo caso, dado $a \in A$, $\pi(a) = a + M \in I$, y por lo tanto $I = A/M$. Esto muestra que A/M es un campo.

Recíprocamente, supongamos que A/M es un campo. Sea I un ideal tal que

$$M \subset I \subset A.$$

Como π es sobreyectiva, $\pi(I)$ es un ideal de A/M, y como A/M es un campo entonces $\pi(I) = \{M\}$ o $\pi(I) = A/M$. En el primer caso, dado $a \in I$, tenemos que $a + M = \pi(a) = M$ de donde $a \in M$. Así $I \subset M$ y como $M \subset I$, concluimos que $M = I$. En el segundo caso, dado $b \in A$, existe $a \in I$ tal que $b + M = \pi(a) = a + M$. Luego $a - b \in M \subset I$ y luego

$$b = a - (a - b) \in I,$$

y por lo tanto $I = A$.

La siguiente proposición es inmediata; la prueba es idéntica a la de la proposición 1.6.12.

Proposición 1.7.9 Sea $f : A \to B$ un homomorfismo de anillos. f es un monomorfismo si y sólo si $\mathrm{Ker}(f) = \{0\}$.

1.8 Anillos de Polinomios

Sea $(A, +, \cdot)$ un anillo. Un *polinomio en una variable* sobre A es una sucesión $(a_0, a_1, \dots, a_n, \dots)$, donde $a_j \in A$ para todo $j \geq 0$ y $a_i \neq 0$ solamente para un número finito de índices. Sea $\mathcal{A}$ el conjunto de polinomios en una variable sobre A. En $\mathcal{A}$ definimos las operaciones $\oplus : \mathcal{A} \times \mathcal{A} \to \mathcal{A}$ y $\odot : \mathcal{A} \times \mathcal{A} \to \mathcal{A}$ por

$$(a_0, a_1, \dots) \oplus (b_0, b_1, \dots) = (a_0 + b_0, a_1 + b_1, \dots);$$

$$(a_0, a_1, \dots) \odot (b_0, b_1, \dots) = (c_0, c_1, \dots);$$

donde

$$c_n = \sum_{k=0}^{n} a_k b_{n-k}.$$

Es fácil comprobar que $(\mathcal{A}, \oplus, \odot)$ es un anillo, donde el elemento neutro de $\oplus$ es $(0, 0, 0, \dots)$, el inverso con respecto a $\oplus$ de $(a_0, a_1, \dots, a_n, \dots)$ es $(-a_0, -a_1, \dots, -a_n, \dots)$, y el elemento neutro de $\odot$ es $(1, 0, 0, \dots)$. Consideremos el subanillo B de A dado por

$$B = \{(a, 0, 0, \dots) : a \in A\},$$

este subanillo es isomorfo a A a través del homomorfismo $f : A \to B$, $f(a) = (a, 0, 0, \dots)$. Por lo tanto identificamos B con A y denotamos $(a, 0, 0, \dots)$ por a (no habrá peligro de confusión). Si además denotamos $x = (0, 1, 0, 0, \dots)$, y las operaciones $\oplus$, $\odot$ simplemente por $+, \cdot$, entonces

$$(a_0, a_1, \dots, a_n, 0, 0, \dots) = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n$$

para todo elemento $(a_0, a_1, \ldots, a_n, 0, 0, \ldots)$ de $\mathcal{A}$. Así que de ahora en adelante denotamos el anillo $(\mathcal{A}, \oplus, \odot)$ por $(A[x], +, \cdot)$.

Definición 1.8.1 Sea $f(x) \in A[x]$, $f(x) \neq 0$. El entero n tal que

$$f(x) = a_0 + a_1 x + \cdots + a_n x^n$$

con $a_n \neq 0$, se denomina *grado de* $f(x)$.

Proposición 1.8.1 Sea D un dominio y sean $f(x), g(x) \in D[x]$, $f(x) \neq 0$ y $g(x) \neq 0$, entonces $f(x) \cdot g(x) \neq 0$ y

$$\text{grado}(f(x) \cdot g(x)) = \text{grado}\, f(x) + \text{grado}\, g(x).$$

Prueba. Si grado $f(x) = m$, grado $g(x) = n$, y escribiendo

$$f(x) = a_0 + a_1 x + \cdots + a_m x^m, \qquad g(x) = b_0 + b_1 x + \cdots + b_n x^n$$

donde $a_m \neq 0, b_m \neq 0$, tenemos que

$$f(x)g(x) = a_0 b_0 + (a_0 b_1 + a_1 b_0)x + \cdots + a_m b_n x^{m+n},$$

y como A es dominio, $a_m b_m \neq 0$. De ahí $f(x) \cdots g(x) \neq 0$ y

$$\text{grado}(f(x) \cdot g(x)) = \text{grado}\, f(x) + \text{grado}\, g(x).$$

Definición 1.8.2 Sea D un dominio y $f(x) \in D[x]$ un polinomio de grado positivo (grado$f(x) \geq 1$). Entonces $f(x)$ se dice *irreducible sobre* D si no puede ser escrito como el producto de dos polinomios en $D[x]$, cada uno de grado positivo.

Proposición 1.8.2 Sea A un anillo. Sea $f(x) \in A[x]$ y sea $g(x) = b_m x^m + \cdots + b_1 x + b_0 \in A[x]$ con b_m inversible en A. Entonces existen únicos $q(x), r(x) \in A[x]$ tales que

$$f(x) = g(x)q(x) + r(x) \text{ con } \begin{cases} \text{grado}\, r(x) < \text{grado}\, g(x) \\ \text{o} \quad r(x) = 0 \end{cases} \tag{1.4}$$

En particular, si A es un campo, basta imponer sobre $g(x)$ la condición $g(x) \neq 0$.

Prueba. Realizamos la prueba por inducción sobre grado $f(x)$. Si grado $f(x) <$ grado $g(x)$, entonces tomamos

$$q(x) = 0, \quad r(x) = f(x).$$

Veamos el otro caso. Sea

$$f(x) = a_n x^n + \cdots + a_1 x + a_0,$$

y $m = \text{grado}\, g(x)$. Entonces

$$f(x) = a_n b_m^{-1} x^{n-m} g(x) + f_1(x),$$

donde $\text{grado}\, f_1(x) < \text{grado}\, f(x)$. Por la hipótesis inductiva,

$$f_1(x) = q_1(x)g(x) + r(x) \quad \text{con} \quad \text{grado}\, r(x) < \text{grado}\, g(x)$$

o $r(x) = 0$. Haciendo

$$q(x) = a_n b_m^{-1} x^{n-m} + q_1(x)$$

tenemos (1.5). La unicidad de $q(x), r(x)$ es fácil de probar. Si

$$f(x) = g(x)q_1(x) + r_1(x) \text{ con } \begin{cases} \text{grado}\, r_1(x) < \text{grado}\, g(x) \\ \text{o } \ r(x) = 0 \end{cases} \tag{1.5}$$

entonces

$$g(x) = (q_1(x) - q(x)) = r(x) - r_1(x).$$

Si $q_1(x) \neq q(x)$, entonces $r(x) \neq r_1(x)$ y

$$\text{grado}(r(x) - r_1(x)) < \text{grado}\, g(x) \leq \text{grado}(g(x)(q_1(x) - q(x))) = \text{grado}(r(x) - r_1(x)),$$

absurdo. Por lo tanto

$$q(x) = q_1(x) \quad \text{y} \quad q_1(x) = q(x).$$

Nota. El polinomio $q(x)$ obtenido en la proposición anterior es llamado *cociente*, mientras que $r(x)$ es llamado *residuo*. Si $r(x) = 0$ decimos que $h(x)$ *divide a* $g(x)$ y denotamos $h(x) | g(x)$.

Definición 1.8.3 Sea $f(x) \in A[x]$ dado, donde A es un anillo. Sean $g(x), h(x) \in A[x]$; decimos que $g(x)$ *es congruente a* $h(x)$ *módulo* $g(x)$ si $f(x)$ divide a $g(x) - h(x)$, y lo denotamos

$$g(x) \equiv h(x) (\bmod f(x)).$$

¿Cuál es la forma del anillo cociente $A[x]/(f(x))$? Si $\text{grado} f(x) = n$, entonces dado $g(x) + (f(x)) \in A[x]/(f(x))$, por el algoritmo de la división de polinomios, $g(x) = f(x)q(x) + r(x)$ donde $r(x) = 0$ o $\text{grado}\, r(x) < n$, esto es $r(x) = a_0 + a_1 x + \cdots + a_{n-1} x^{n-1}$. Luego un sistema de representantes para $A[x]/(f(x))$ es

$$\{a_0 + a_1 x + \cdots + a_{n-1} x^{n-1} : \ a_j \in A, \ 1 \leq j < n\}.$$

La siguiente proposición es inmediata:

Proposición 1.8.3 Si en $A[x]$ consideramos el ideal generado por $f(x) \in A[x]$, $I = (f(x))$, entonces la relación de equivalencia correspondiente al anillo cociente $A[x]/(f(x))$ es precisamente la relación de congruencia de polinomios, indicada arriba.

La siguiente proposición deja ver una clara similitud entre las propiedades de $\mathbb{Z}$ y las de $\mathbb{F}[x]$ ($\mathbb{F}$ es un campo). Esto en realidad se debe a que ambos son *dominios euclidianos*; ver [7].

Proposición 1.8.4 Dados dos polinomios no nulos $f(x), g(x) \in \mathbb{F}[x]$ existe un único polinomio $d(x) \in \mathbb{F}[x]$ (salvo por la multiplicación por un escalar positivo) tal que $d(x)|f(x)$, $d(x)|g(x)$ y tal que si $e(x) \in \mathbb{F}[x]$ cumple que $e(x)|f(x)$, $e(x)|g(x)$ entonces $e(x)|d(x)$. Además este polinomio se escribe en la forma

$$d(x) = r(x)f(x) + s(x)g(x),$$

con $r(x), s(x) \in \mathbb{F}[x]$. Este polinomio es conocido como el *máximo común divisor* de los polinomios $f(x)$ y $g(x)$; lo denotaremos $\text{mcd}(f(x), g(x))$.

Prueba. Sea

$$I(f(x), g(x)) = \{r(x)f(x) + s(x)g(x) : r(x), s(x) \in \mathbb{F}[x]\},$$

de este conjunto escogemos un elemento $d(x) = r_0(x)f(x) + s_0(x)g(x) \in I(f(x), g(x))$ de grado mínimo entre los elementos no nulos de $I(f(x), g(x))$. El resto de la demostración es idéntico a la demostración del teorema 1.4.1.

Proposición 1.8.5 Sea $\mathbb{F}$ un campo. Si $f(x) \in \mathbb{F}[x]$ es irreducible sobre $\mathbb{F}$, entonces $\mathbb{F}[x]/(f(x))$ es un campo.

Prueba. Por el criterio dado por el teorema 1.7.8, basta probar que el ideal $M = (f(x))$ es maximal en $\mathbb{F}[x]$. En efecto, supongamos que I es un ideal en $\mathbb{F}[x]$ tal que

$$M \subset I \subset \mathbb{F}[x].$$

Si $M \neq I$, escogemos un elemento $g(x) \in I \setminus M$. Consideremos $d(x) = \text{mcd}(f(x), g(x))$. Si grado $d(x) \geq 1$, como $f(x)$ es irreducible y $d(x)|f(x)$ entonces $d(x) = f(x)$ (salvo por una constante), de donde $d(x)|g(x)$, absurdo. Por lo tanto, $d(x) = 1$. Existen entonces elementos $r(x), s(x) \in \mathbb{F}[x]$ tales que

$$r(x)f(x) + s(x)g(x) = 1,$$

de donde $1 \in M + (g(x))$ y

$$\mathbb{F}[x] \subset M + (g(x)) \subset I \subset \mathbb{F}[x],$$

y por lo tanto $I = \mathbb{F}[x]$. Por lo tanto M es maximal.

Dado un polinomio $f(x) = \sum_{i=0}^{n} a_i x^i \in A[x]$, consideremos la *función polinomial* asociada $\tilde{f} : A \to A$, definida por $\tilde{f}(b) = \sum_{i=0}^{n} a_i b^i$, $b \in A$. Si no hay peligro de confusión indicaremos $f(b)$ en vez de $f(x)$ para denotar tanto al elemento de A como al de $A[x]$.

Definición 1.8.4 Sean A un anillo y $f(x) \in A[x]$. Decimos que $a \in A$ es una *raíz de* $f(x)$ si $f(a) = 0$.

Ahora relacionamos las raíces de $f(x)$ con sus factores en $A[x]$:

Proposición 1.8.6 Sean A un anillo, $f(x) \in A[x]$ y $a \in A$. Entonces $f(a) = 0$ si y sólo si existe $g(x) \in A[x]$ tal que

$$f(x) = (x - a) \cdot g(x).$$

Prueba. Consideramos $f(x) \neq 0$. Por el algoritmo de división 1.8.2, existen $g(x), r(x) \in A[x]$ tales que

$$f(x) = g(x)(x - a) + r(x),$$

donde grado $r(x) < 1$ o $r(x) = 0$, esto es, $r(x) = c \in A$ (una constante). Evaluando la expresión anterior en $x = a$,

$$f(a) = g(a)(a - a) + c = c,$$

de donde $f(a) = 0$ si y sólo si $c = 0$, i.e., si y sólo si $f(x) = (x - a)g(x)$.

Definición 1.8.5 Sean A un anillo, $f(x) \in A[x]$, $a \in A$. Decimos que a es una *raíz de* $f(x)$ *con multiplicidad* $e \geq 1$ si existe $h(x) \in A[x]$ tal que

$$f(x) = (x - a)^e \cdot h(x) \text{ con } h(a) \neq 0.$$

Para un dominio, tenemos el

Teorema 1.8.7 Sea D un dominio y $f(x) \in D[x]$, $f(x) \neq 0$ un polinomio de grado n. Entonces el número de raíces de $f(x)$ (contando las multiplicidades), no puede exceder n.

Prueba. Haremos la prueba por inducción sobre $n = \text{grado}\, f(x)$. Si $n = 0$ o $n = 1$ el resultado es trivial. Supongamos que $f(x)$ tiene más de n raíces (incluidas las multiplicidades). Escogemos una raíz a de $f(x)$, y escribimos

$$f(x) = (x - a)^e g(x)$$

donde $e \geq 1$, $g(a) \neq 0$. Pero entonces $g(x)$ es un polinomio de grado $n - e$ con más de $n - e$ raíces (el número de raíces, contando multiplicidades, de dos polinomios se suma cuando se multiplican), contradiciendo la hipótesis inductiva.

1.9 Campos finitos

Si F es un campo finito de n elementos, lo denotamos por $\mathbb{F}_n$. Así al campo $\mathbb{Z}_p$ para p primo se le denota indiferentemente por $\mathbb{F}_p$ o $\mathbb{Z}_p$.

Teorema 1.9.1 Sea $\mathbb{F}_n$ un campo finito. Entonces existe un primo p y $k \in \mathbb{N}$ tales que $n = p^k$.

Prueba. La característica de $\mathbb{F}_n$ debe ser no nula (de otro modo $\mathbb{F}_n$ tendría infinitos elementos), luego es un primo $p = \text{car}(\mathbb{F}_n)$. El conjunto

$$G = \{j \cdot 1 \,/\, 0 \leq j < p\} \quad (\text{con } 1 \in \mathbb{F}_n)$$

es un subcampo de $\mathbb{F}_n$, isomorfo a $\mathbb{Z}_p$. Luego $\mathbb{F}_n$ puede ser visto como un espacio vectorial sobre G, y por consiguiente sobre $\mathbb{Z}_p$. Como $\mathbb{F}_n$ es finitamente generado (ya que es finito), podemos escoger una base $\{v_1, \ldots, v_k\} \subset \mathbb{F}_n$, de donde todo $v \in \mathbb{F}_n$ se escribe de manera única como

$$v = \alpha_1 v_1 + \cdots + \alpha_k v_k \text{ con } \alpha_i \in G,\ 1 \leq i \leq k.$$

De ahí es obvio que $n = p^k$, ya que la aplicación $\varphi : F \to G^n$ definida por $\varphi(v) = (\alpha_1, \ldots, \alpha_k)$ es una biyección.

Teorema 1.9.2 Sea $\mathbb{F}_n$ un campo finito. Entonces $\mathbb{F}_n^*$ es cíclico.

Prueba. Usaremos el criterio dado por el teorema 1.6.8. Sea $G = \mathbb{F}_n^* = \mathbb{F}_n \setminus \{0\}$. Es claro que $\exp G \leq |G|$ siempre. Ahora consideremos el polinomio

$$f(x) = x^{\exp G} - 1 \in \mathbb{F}_n[x];$$

vemos que todo $a \in G$ es una raíz de f. Como $f(x)$ tiene a lo más $\exp G$ raíces, entonces $|G| \leq \exp G$. Por lo tanto G es cíclico.

1.10 Restos cuadráticos

Definición 1.10.1 Sean p, a enteros con p primo. Si la congruencia

$$x^2 \equiv a(\bmod p)$$

tiene una solución, decimos que a es un *resto cuadrático* $\bmod p$, y si no la tiene, decimos que es un *resto no cuadrático.*

El símbolo de Legendre

Definición 1.10.2 Sea p un primo impar. Si $a \neq 0(\bmod p)$ definimos el *símbolo de Legendre*

$$\left(\frac{a}{p}\right) = \begin{cases} +1, & \text{si } a \text{ es un resto cuadrático } \bmod p, \\ -1, & \text{si } a \text{ es un resto no cuadrático } \bmod p, \end{cases}$$

y si $a \equiv 0(\bmod p)$ definimos $\left(\frac{a}{p}\right) = 0$.

Es claro de la definición que si $a \equiv b(\bmod p)$, entonces $\left(\frac{a}{p}\right) = \left(\frac{b}{p}\right)$.

Teorema 1.10.1 Sea p un primo impar. Entonces cada sistema residual reducido contiene exactamente $(p-1)/2$ restos cuadráticos y $(p-1)/2$ restos no cuadráticos $\bmod p$.

Prueba. En efecto, los restos cuadráticos son precisamente las clases residuales que contienen a los números

$$1^2, 2^2, 3^2, \ldots, \left(\frac{p-1}{2}\right)^2.$$

Estos números son distintos módulo p, ya que si $x^2 \equiv y^2(\bmod p)$ con $1 \leq x \leq (p-1)/2$, $1 \leq y \leq (p-1)/2$ entonces

$$(x-y)(x+y) \equiv 0(\bmod p)$$

y dado que $1 < x+y < p$, entonces $x - y \equiv 0(\bmod p)$ y por lo tanto $x \equiv y(\bmod p)$. Además, $(p-k)^2 \equiv k^2(\bmod p)$, así que todo resto cuadrático módulo p es alguno de los indicados arriba.

Teorema 1.10.2 (Criterio de Euler) Sea p un primo impar. Entonces

$$\left(\frac{a}{p}\right) \equiv a^{(p-1)/2}(\bmod p),$$

para todo entero a.

Prueba. Si $a \equiv 0(\bmod p)$ el resultado es trivial. Supongamos que $\left(\frac{a}{p}\right) = 1$, i.e. existe un entero b tal que $b^2 \equiv a(\bmod p)$. Entonces

$$a^{(p-1)/2} \equiv (b^2)^{(p-1)/2} \equiv b^{p-1} \equiv 1 \equiv \left(\frac{a}{p}\right) (\bmod p).$$

Supongamos ahora que $\left(\frac{a}{p}\right) = -1$. Consideremos el polinomio $f(x) = x^{(p-1)/2} - 1$ en $\mathbb{Z}_p$; como el grado de $f(x)$ es $(p-1)/2$, la congruencia

$$f(x) \equiv 0(\bmod p)$$

tiene a lo más $(p-1)/2$ soluciones, que son justamente los restos cuadráticos $\bmod p$. Por lo tanto los restos no cuadráticos no son soluciones de la congruencia, i.e.

$$a^{(p-1)/2} \not\equiv 1(\bmod p). \tag{1.6}$$

Además $(a^{(p-1)/2})^2 \equiv a^{p-1} \equiv 1(\bmod p)$ implica que $a^{(p-1)/2} \equiv \pm 1(\bmod p)$ (pues la congruencia $x^2 \equiv 1(\bmod p)$ tiene exactamente dos soluciones, $x \equiv \pm 1(\bmod p)$), y de (1.6),

$$a^{(p-1)/2} \equiv -1 \equiv \left(\frac{a}{p}\right) (\bmod p).$$

Teorema 1.10.3 El símbolo de Legendre $\left(\frac{a}{p}\right)$ satisface:

$$\left(\frac{ab}{p}\right) = \left(\frac{a}{p}\right)\left(\frac{b}{p}\right),$$

para cualesquiera a, b enteros.

Prueba. En efecto,

$$\left(\frac{ab}{p}\right) \equiv (ab)^{(p-1)/2} \equiv a^{(p-1)/2} b^{(p-1)/2} \equiv \left(\frac{a}{p}\right)\left(\frac{b}{p}\right) (\bmod p),$$

y el valor absoluto de la diferencia de los términos en los extremos de la ecuación es ≤ 2, de donde son iguales.

Teorema 1.10.4 Para p primo impar

$$\left(\frac{-1}{p}\right) = (-1)^{(p-1)/2} = \begin{cases} 1, & \text{si } p \equiv 1(\bmod 4), \\ -1, & \text{si } p \equiv 3(\bmod 4). \end{cases}$$

Prueba. En efecto,

$$\left(\frac{-1}{p}\right) \equiv (-1)^{(p-1)/2} (\mathrm{mod}\, p),$$

y la diferencia de los términos tiene valor absoluto ≤ 2, de donde son iguales.

Observación. Sea a entero inversible módulo p. Para cada j, $1 \leq j \leq (p-1)/2$ escribimos

$$a \cdot j = \varepsilon_j m_j (\mathrm{mod}\, p),$$

donde $\varepsilon_j = \varepsilon_j(a) = \pm 1$, $1 \leq m_j \leq (p-1)/2$. Si $m_i = m_j$ entonces $a \cdot i \equiv a \cdot j (\mathrm{mod}\, p)$, en cuyo caso $i = j$, o bien $a \cdot i \equiv -a \cdot j (\mathrm{mod}\, p)$, lo cual es imposible pues $1 < i + j < p$. Por lo tanto los m_i son distintos, y

$$\{m_1, m_2, \ldots, m_{\frac{p-1}{2}}\} = \{1, 2, \ldots, (p-1)/2\}$$

y por el criterio de Euler,

$$\begin{aligned}
\left(\frac{a}{p}\right) &\equiv a^{\frac{p-1}{2}} \equiv \frac{(a \cdot 1)(a \cdot 2) \ldots (a \cdot \frac{p-1}{2})}{1 \cdot 2 \cdot \ldots \cdot \frac{p-1}{2}} (\mathrm{mod}\, p) \\
&\equiv \frac{\varepsilon_1 \varepsilon_2 \cdots \varepsilon_{\frac{p-1}{2}} m_1 m_2 \ldots m_{\frac{p-1}{2}}}{1 \cdot 2 \cdot \ldots \cdot \frac{p-1}{2}} (\mathrm{mod}\, p) \\
&\equiv \varepsilon_1 \varepsilon_2 \cdots \varepsilon_{\frac{p-1}{2}} (\mathrm{mod}\, p)
\end{aligned}$$

y por lo tanto $\left(\frac{a}{p}\right) = (-1)^m$, donde

$$m = \#\{j : 1 \leq j \leq (p-1)/2, \varepsilon_j = -1\}.$$

Esta observación nos permite demostrar el siguiente teorema.

Teorema 1.10.5 Sea p primo impar,

$$\left(\frac{2}{p}\right) = (-1)^{(p^2-1)/8} = \begin{cases} 1, & \text{si } p \equiv \pm 1 (\mathrm{mod}\, 8), \\ -1, & \text{si } p \equiv \pm 3 (\mathrm{mod}\, 8). \end{cases}$$

Prueba. Utilizamos la observación anterior con $a = 2$. Si $p \equiv 1 (\mathrm{mod}\, 4)$, escribimos $\frac{p-1}{2} = 2k$. Si $1 \leq j \leq k$, entonces $1 \leq 2j \leq \frac{p-1}{2}$ y si $k+1 \leq j \leq 2k$, entonces $\frac{p-1}{2} < 2j \leq p-1$. Por lo tanto

$$\left(\frac{a}{p}\right) = (-1)^k = \begin{cases} 1, & p \equiv 1 (\mathrm{mod}\, 8), \\ -1, & p \equiv 5 (\mathrm{mod}\, 8). \end{cases}$$

Si $p \equiv 3(\mathrm{mod}\,4)$, escribimos $\dfrac{p-1}{2} = 2k+1$. Si $1 \leq j \leq k$, entonces $1 \leq 2j \leq \dfrac{p-1}{2}$, y si $k+1 \leq j \leq 2k+1$ entonces $\dfrac{p-1}{2} < 2j \leq p-1$. Por lo tanto

$$\left(\frac{a}{p}\right) = (-1)^{k+1} = \begin{cases} 1, & p \equiv 3(\mathrm{mod}\,8), \\ -1, & p \equiv 7(\mathrm{mod}\,8). \end{cases}$$

Teorema 1.10.6 (Ley de reciprocidad cuadrática) Si p, q son primos impares distintos, entonces

$$\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = (-1)^{(p-1)(q-1)/4}.$$

Prueba. Utilizaremos nuevamente la notación dada antes, con $a = q$. Para cada $j \in P$,

$$P = \{1, 2, \ldots, (p-1)/2\}$$

tenemos que $\varepsilon_j = -1$ si y sólo si existe $k \in P$ con

$$q \cdot j \equiv -j(\mathrm{mod}\,p),$$

i.e. si y sólo si

$$q \cdot j = -k + py,$$

para algún $y \in \mathbb{Z}$. Haciendo $k = py - qj$ tenemos que $-(p-1)/2 \leq -k < 0$. Luego $\varepsilon_j = -1$ si y sólo si existe $y \in \mathbb{Z}$ tal que

$$-(p-1)/2 \leq qj - py < 0.$$

Tal y debe pertenecer a

$$Q = \{1, 2, \ldots, (q-1)/2\}.$$

En efecto, de $0 < py - qj < (p-1)/2$ tenemos que

$$0 < qj < py \leq qj + (p-1)/2 \leq q(p-1)/2 + (p-1)/2 < qp/2 + p/2 = p\frac{q+1}{2},$$

de donde

$$0 < y < \frac{q+1}{2},$$

esto es

$$1 \leq y \leq \frac{q+1}{2} - 1 = \frac{q-1}{2}.$$

Además, para cada j, el entero y es único. Si

$$-(p-1)/2 \leq qj - py_1 < 0$$

y como $0 < py - qj \leq (p-1)/2$, sumando las desigualdades

$$-(p-1)/2 < p(y - y_1) < (p-1)/2,$$

de donde

$$|p(y - y_1)| \leq \frac{p-1}{2} < p,$$

esto es $|y - y_1| < 1$ de donde $y = y_1$. Por lo tanto $\left(\frac{q}{p}\right) = (-1)^m$ donde $m = \#X$ y

$$X = \{(x, y) \in P \times Q : -(p-1)/2 \leq qx - py < 0\}.$$

Análogamente, $\left(\frac{p}{q}\right) = (-1)^n$, donde $n = \#Y$ e

$$Y = \{(x, y) \in P \times Q : 0 < qx - py \leq (q-1)/2\}.$$

De ahí, y dado que $qx - py$ nunca se anula, entonces $\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = (-1)^k$ donde $k = m + n = \#Z$,

$$Z = \{(x, y) \in P \times Q : -(p-1)/2 \leq qx - py \leq (p-1)/2\}.$$

Tenemos que $k = |C| - |A| - |B|$ donde $C = P \times Q$,

$$A = \{(x, y) \in C : qx - py < -(p-1)/2\}, \qquad B = \{(x, y) \in C : qx - py > (q-1)/2\}.$$

Como $\#C = (p-1)(q-1)/4$, basta mostrar que $\#A = \#B$. La aplicación $f : C \to C$ dada por $f(x, y) = ((p+1)/2 - x, (q+1)/2 - y)$ define una biyección entre A y B; lo cual completa la prueba.

El símbolo de Jacobi

Introducimos una generalización del símbolo de Legendre que nos ayudará más fácilmente a determinar si un número compuesto dado es un número de Legendre.

Definición 1.10.3 Si a, n son enteros con n positivo impar, $n > 1$ tal que su descomposición en factores primos es

$$n = \prod_{i=1}^{r} p_i^{a_i}$$

definimos el *símbolo de Jacobi* $\left(\frac{a}{n}\right)$ por

$$\left(\frac{a}{n}\right) = \prod_{i=1}^{r} \left(\frac{a}{p_i}\right)^{a_i}$$

donde $\left(\frac{a}{p_i}\right)$ es el símbolo de Legendre. Definimos además $\left(\frac{a}{1}\right) = 1$.

Observe que $\left(\frac{a}{n}\right)$ toma, al igual que el símbolo de Legendre, los valores -1, 1 y 0, y $\left(\frac{a}{n}\right) = 0$ si y sólo si $(a,n) > 1$. Además, si la congruencia $x^2 \equiv a(\bmod n)$ posee solución entonces $\left(\frac{a}{n}\right) = 1$, aunque el recíproco no es cierto. Observaremos cómo algunas propiedades del símbolo de Legendre se conservan para el símbolo de Jacobi.

Teorema 1.10.7 Si m, n son enteros positivos impares, y a, b enteros cualesquiera,

(i) $\left(\frac{a}{m}\right)\left(\frac{b}{m}\right) = \left(\frac{ab}{m}\right)$;

(ii) $\left(\frac{a}{m}\right)\left(\frac{a}{n}\right) = \left(\frac{a}{mn}\right)$;

(iii) $\left(\frac{a}{m}\right) = \left(\frac{b}{m}\right)$ si $a \equiv b(\bmod m)$;

(iv) $\left(\frac{a^2 b}{m}\right) = \left(\frac{b}{m}\right)$ si $(a,m) = 1$.

Teorema 1.10.8 Si n es un entero positivo impar,

$$\left(\frac{-1}{n}\right) = (-1)^{(n-1)/2}$$

y

$$\left(\frac{2}{n}\right) = (-1)^{(n^2-1)/8}.$$

Prueba. El caso $n = 1$ es trivial. Si $n > 1$ escribimos $n = p_1 \cdots p_m$ donde los p_i son primos no necesariamente distintos. Entonces

$$n = \prod_{i=1}^{m}(1 + p_i - 1) = 1 + \sum_{i=1}^{m}(p_i - 1) + \sum_{i \neq j}(p_i - 1)(p_j - 1) + \cdots$$

Es obvio que $p_i - 1 \equiv 0(\bmod 2)$, de donde todas las sumas arriba, salvo la primera, son múltiplos de 4, y

$$n \equiv 1 + \sum_{i=1}^{m}(p_i - 1)(\bmod 4).$$

Entonces

$$\frac{1}{2}(n-1) \equiv \sum_{i=1}^{m}\frac{1}{2}(p_i - 1)(\bmod 2), \tag{1.7}$$

y por lo tanto

$$\left(\frac{-1}{n}\right) = \prod_{i=1}^{m}\left(\frac{-1}{p_i}\right) = \prod_{i=1}^{m}(-1)^{(p_i-1)/2} = (-1)^{(n-1)/2}.$$

Para probar la segunda identidad desarrollamos

$$n^2 = \prod_{i=1}^{m}(1 + p_i^2 - 1) = 1 + \sum_{i=1}^{m}(p_i^2 - 1) + \sum_{i \neq j}(p_i^2 - 1)(p_j^2 - 1) + \cdots,$$

como p_i impar, entonces $p_i^2 - 1 \equiv 0 (\bmod 8)$ y entonces

$$n^2 \equiv 1 + \sum_{i=1}^{m}(p_i^2 - 1) (\bmod 64),$$

de donde la identidad arriba es también módulo 16 y

$$\frac{1}{8}(n^2 - 1) \equiv \sum_{i=1}^{m} \frac{1}{8}(p_i^2 - 1) (\bmod 2);$$

por lo tanto

$$\left(\frac{2}{n}\right) = \prod_{i=1}^{m}\left(\frac{2}{p_i}\right) = \prod_{i=1}^{m}(-1)^{(p_i^2-1)/8} = (-1)^{(n^2-1)/8}.$$

Teorema 1.10.9 (Ley de reciprocidad cuadrática para el símbolo de Jacobi) Si m, n son enteros positivos impares, $(m, n) = 1$, entonces

$$\left(\frac{m}{n}\right)\left(\frac{n}{m}\right) = (-1)^{(m-1)(n-1)/4}.$$

Prueba. En efecto, escribiendo $m = p_1 \cdots p_r$, $n = q_1 \cdots q_s$ con p_i, q_j primos (no necesariamente distintos), y aplicando la ley de reciprocidad cuadrática para los símbolos $\left(\frac{p_i}{q_j}\right)$,

$$\left(\frac{p}{q}\right)\left(\frac{q}{p}\right) = \prod_{i=1}^{r}\prod_{j=1}^{s}\left(\frac{p_i}{q_j}\right)\left(\frac{q_j}{p_i}\right) = (-1)^t,$$

donde $t = \sum_{i=1}^{r}\sum_{j=1}^{s}\frac{1}{2}(p_i - 1)\frac{1}{2}(q_j - 1)$. Pero

$$t = \sum_{i=1}^{r}\frac{1}{2}(p_i - 1)\sum_{j=1}^{s}\frac{1}{2}(q_j - 1),$$

y de la ecuación (1.7),

$$\sum_{i=1}^{r}\frac{1}{2}(p_i - 1) \equiv \frac{1}{2}(m - 1) (\bmod 2)$$

y

$$\sum_{j=1}^{s}\frac{1}{2}(q_j - 1) \equiv \frac{1}{2}(n - 1) (\bmod 2)$$

de donde

$$t \equiv \frac{1}{4}(m-1)(n-1)(\bmod 2)$$

y se sigue el resultado.

A partir de las propiedades obtenidas, deducimos un algoritmo eficiente para calcular el símbolo de Jacobi de un entero a módulo n.

Algoritmo 1.10.1 Dados dos enteros m, n positivos, con n impar, este algoritmo calcula el símbolo de Jacobi $\left(\frac{m}{n}\right)$.

Entrada: $m, n > 0$ enteros con n impar
si $(m, n) \neq 1$
    $J = 0$
sino
    $J = 1$
    mientras $(m \neq 1)$
        $m = m \bmod n$
        Escribir $m = 2^s t$, con t impar
        [cálculo de $(\frac{2}{n})^s$]
        si $(s \bmod 2 = 1 \wedge (n \bmod 8 = 3 \vee n \bmod 8 = 5))$
            $J = -J$
        [ley de reciprocidad cuadrática]
        si $(t \bmod 4 = 3 \wedge n \bmod 4 = 3)$
            $J = -J$
        $m = n$
        $n = t$
Salida: $J = (\frac{m}{n})$

El costo total del algoritmo para calcular el símbolo de Jacobi es $O(\log^3 n)$: el costo de todos los pasos es conocido salvo la factorización $m = 2^s t$; observe que como $2^s \leq m$, entonces $s \leq \log_2 m$, y se realizan s divisiones entre 2, cada una de costo $O(\log n)$; esto es, la factorización tiene complejidad $O(\log^2 n)$. El número de iteraciones es menor o igual a las iteraciones realizadas por el algoritmo euclidiano, $O(\log n)$.

1.11 Raíces primitivas

Definición 1.11.1 Sean a, n enteros con $n > 1$, $(a, n) = 1$. El menor entero positivo f tal que

$$a^f \equiv 1 (\bmod n)$$

se denomina *exponente de a módulo n*, y se denota por $\exp_n(a)$.

Es fácil ver que $\exp_n(a)$ no es nada menos que el orden $o(\overline{a})$ del elemento $\overline{a} \in \mathbb{Z}_n^*$. Por el teorema de Euler, todo entero a tal que $(a, n) = 1$ satisface $a^{\varphi(n)} \equiv 1 (\bmod n)$ (y por ende $\exp_n(a) | \varphi(n)$).

Definición 1.11.2 Sean a, n enteros con $n > 1$ y $(a, n) = 1$. Decimos que a es una *raíz primitiva módulo n* si $\exp_n(a) = \varphi(n)$.

Una raíz primitiva corresponde entonces precisamente a un entero a tal que $\overline{a}$ es un generador de $\mathbb{Z}_n^*$. Es importante el saber para qué valores de n existen estas raíces primitivas. Por el teorema 1.9.2, para p primo el subgrupo multiplicativo $\mathbb{Z}_p^*$ del campo $\mathbb{Z}_p$ es cíclico. Esto se traduce en el siguiente

Teorema 1.11.1 Existen raíces primitivas módulo p para p primo.

Ahora buscamos extender este resultado para otros valores de n.

Lema 1.11.1 Sea p primo y $a \in \mathbb{Z}$ una raíz primitiva módulo p. Entonces a o $a + p$ es una raíz primitiva módulo p^2.

Prueba. Veamos que

$$(a+p)^p = \sum_{i=0}^{p} \binom{p}{i} a^i p^{p-i} \equiv \binom{p}{p-1} a^{p-1} p + a^p \equiv a^p (\bmod p^2).$$

Sin pérdida de generalidad podemos suponer que $a^p \equiv a (\bmod p^2)$ (sino tomamos $a+p$ en vez de a); luego $a^{p-1} \not\equiv 1 (\bmod p^2)$, de donde $\exp_{p^2}(a) \neq p - 1$. Como $p - 1 = \exp_p(a) | \exp_{p^2}(a)$ y $\exp_{p^2}(a) | \varphi(p^2) = p(p-1)$, concluimos que $\exp_{p^2}(a) = p(p-1)$.

Lema 1.11.2 Sea p un primo impar y a una raíz primitiva módulo p^2, entonces a es una raíz primitiva módulo p^k para todo $k > 2$, $k \in \mathbb{Z}$.

Prueba. Tenemos que $a^{p-1} \equiv 1(\bmod p)$ pero $a^{p-1} \equiv 1(\bmod p^2)$, de donde $a^{p-1} = 1 + b_0 p$ con $b_0 \not\equiv 0(\bmod p)$. Veamos, por inducción, que

$$a^{p^j(p-1)} = 1 + b_j p^{j+1} \quad \text{con} \quad b_j \equiv b_0(\bmod p). \tag{1.8}$$

En efecto,

$$\begin{aligned} a^{p^{j+1}(p-1)} &= (a^{p^j(p-1)})^p = (1 + b_j p^{j+1})^p = \sum_{i=0}^{p} \binom{p}{i} (b_j p^{j+1})^i \\ &= 1 + p b_j p^{j+1} + \sum_{i=2}^{p} \binom{p}{i} b_j^i p^{ij+i} \\ &= 1 + b_j \left(1 + \sum_{i=2}^{p} \binom{p}{i} b_j^{i-1} p^{(i-1)j+(i-2)} \right) p^{j+2} = 1 + b_{j+1} p^{j+2}, \end{aligned}$$

donde $b_{j+1} = b_j c_j$,

$$c_j = 1 + \sum_{i=2}^{p} \binom{p}{i} b_j^{i-1} p^{(i-1)j+(i-2)}.$$

Ahora veamos que $c_j \equiv 1(\bmod p)$: esto es trivial si $j > 0$ (por los exponentes de p en la suma), y si $j = 0$, $c_j \equiv 1 + \binom{p}{2} b_j \equiv 0(\bmod p)$ pues p es impar. Por lo tanto, $b_{j+1} \equiv b_j(\bmod p)$ y se cumple (1.8). Para $j = k - 2$, por el proceso inductivo,

$$a^{p^{k-2}(p-1)} \not\equiv 1(\bmod p^k) \quad \text{y} \quad p^{k-2}(p-1) = \exp_{p^{k-1}}(a).$$

Como $\exp_{p^{k-1}}(a) | \exp_{p^k}(a)$, y $\exp_{p^k}(a) | \varphi(p^k) = p^{k-1}(p-1)$ y por lo anterior, tenemos que $\exp_{p^k}(a) = p^{k-1}(p-1)$, lo que queríamos.

Tenemos entonces el resultado principal de la sección:

Teorema 1.11.2 Un entero $n > 1$ admite una raíz primitiva módulo n si $n = 2$, $n = 4$ o $n = p^k$, $n = 2p^k$ donde p es un primo impar, $k \in \mathbb{N}$.

Prueba. En realidad, si n no es de una de las formas mencionadas, no existe raíz primitiva módulo n. La prueba de la no existencia para los otros casos de n se puede encontrar en [3], nosotros no la necesitaremos.

Los casos $n = 2$ y $n = 4$ se verifican directamente. La existencia de una raíz primitiva módulo $n = p^k$ (p primo impar) se sigue de los dos lemas anteriores. Para el caso $n = 2p^k$, consideremos una raíz primitiva a módulo p^k, entonces a o $a + p^k$ (el que sea impar) será una raíz primitiva módulo $2p^k$ pues $\varphi(2p^k) = \varphi(p^k)$.

Cálculo de índices

Si n tiene una raíz primitiva módulo g, entonces los elementos $1, g, g^2, \ldots, g^{\varphi(n)-1}$ forman un sistema residual reducido módulo n. Si (a, n) entonces existe un único entero k, con $0 \leq k \leq \varphi(n) - 1$ tal que

$$a \equiv g^k (\bmod n).$$

Este entero es conocido como *índice de a en la base g módulo n*, y lo denotamos por $k = \text{ind}_g a$, o simplemente $\text{ind}\, a$ cuando la raíz primitiva g está prefijada.

El índice goza de las siguientes propiedades, las que le dan el nombre de "logaritmo discreto":

Teorema 1.11.3 Sea g una raíz primitiva módulo n. Si $(a, n) = (b, n) = 1$, entonces

(i) $\text{ind}\, ab \equiv \text{ind}\, a + \text{ind}\, b (\bmod \varphi(n))$;

(ii) $\text{ind}\, a^m \equiv m\, \text{ind}\, a (\bmod \varphi(n))$ si $m \geq 1$;

(iii) $\text{ind}\, 1 = 0$, $\text{ind}\, g = 1$;

(iv) $\text{ind}\, -1 = \varphi(n)/2$ si $n > 2$;

(v) si g' es también una raíz primitiva módulo n entonces

$$\text{ind}_g a = \text{ind}_{g'} a \cdot \text{ind}_g g' (\bmod \varphi(n)).$$

Ahora veremos algunas aplicaciones de la teoría de índices:

Congruencias lineales

Supongamos que n tiene una raíz primitiva y que $(a, n) = (b, n) = 1$. Entonces la congruencia lineal

$$ax \equiv b (\bmod n) \tag{1.9}$$

es equivalente a la congruencia

$$\text{ind}\, a + \text{ind}\, x \equiv \text{ind}\, b (\bmod \varphi(n)).$$

Luego la única solución de (1.9) satisface

$$\text{ind}\, x \equiv \text{ind}\, b - \text{ind}\, a (\bmod \varphi(n)).$$

Congruencias binomiales

Una *congruencia binomial* es una congruencia de la forma

$$x^m \equiv a(\bmod n). \tag{1.10}$$

Si n tiene una raíz primitiva y si $(a, n) = 1$, entonces la congruencia (1.10) es equivalente a

$$m \cdot \operatorname{ind} x \equiv \operatorname{ind} a(\bmod \varphi(n)),$$

congruencia lineal en $\operatorname{ind} x$. Tal congruencia tiene solución si y sólo si $d = (m, \varphi(n))|\operatorname{ind} a$, en cuyo caso tiene d soluciones (ver teorema 1.5.9).

1.12 Fracciones Continuas

Definición 1.12.1 Una *fracción continua finita* (denotada f.c.f.) es una sucesión de números reales $\langle a_0; a_1, \ldots, a_n\rangle$ tal que $a_i > 0$, para $1 \leq i \leq n$. El número n será llamado *longitud* de la f.c.f.

El número

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ddots + \cfrac{1}{a_n}}}}$$

se denomina el *valor* de la f.c.f y se le denota por $[a_0; a_1, \ldots, a_n]$.

Lema 1.12.1 Si $\langle a_0; a_1, \ldots, a_n\rangle$ es una f.c.f, entonces

(i) $[a_0; a_1, \ldots, a_n] = a_0 + [0; a_1, \ldots, a_n]$.

(ii) $[0; a_1, \ldots, a_n] = \dfrac{1}{[a_1; a_2, \ldots, a_n]}$.

(iii) $[a_0; a_1, \ldots, a_n] = a_0 + \dfrac{1}{[a_1; a_2, \ldots, a_n]}$.

(iv) $[a_0; a_1, \ldots, a_n] = [a_0; a_1, \ldots, a_{k-1}, [a_k; a_{k+1}, \ldots, a_n]]$ si $1 \leq k < n$.

(v) Si $a_i \geq 1$, $1 \leq i \leq n$ entonces $0 < [0; a_1, \ldots, a_n] \leq 1$, y

$$[0; a_1, \ldots, a_n] = 1 \quad \text{si y sólo si} \quad n = 1 \text{ y } a_1 = 1.$$

Prueba. Probemos (v). En efecto,

$$0 < [0; a_1, \ldots, a_n] = \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{\ddots + \cfrac{1}{a_n}}}} \leq \frac{1}{a_1} \leq 1.$$

Si $n \geq 2$,

$$[0; a_1, \ldots, a_n] = \frac{1}{a_1 + [0; a_2, \ldots, a_n]} < 1.$$

Definimos los polinomios $P_{-1} = 1$, $Q_{-1} = 0$, $P_0(x_0) = x_0$, $Q_0(x_0) = 1$,

$$\begin{aligned} P_{k+1}(x_0, \ldots, x_{k+1}) &= x_{k+1}P_k(x_0, \ldots, x_k) + P_{k-1}(x_0, \ldots, x_{k-1}), \\ Q_{k+1}(x_0, \ldots, x_{k+1}) &= x_{k+1}Q_k(x_0, \ldots, x_k) + Q_{k-1}(x_0, \ldots, x_{k-1}). \end{aligned}$$

para todo $k \geq 0$.

Lema 1.12.2 Se cumple

$$P_{k+1}Q_k - Q_{k+1}P_k = (-1)^k, \tag{1.11}$$

$$P_{k+2}Q_k - Q_{k+2}P_k = (-1)^k x_{k+2}, \tag{1.12}$$

para todo $k \geq -1$.

Prueba. Por inducción: Tenemos

$$\begin{aligned} P_0Q_{-1} - Q_0P_{-1} &= -1, \\ P_1Q_{-1} - Q_1P_{-1} &= (-1)x_1 \end{aligned}$$

por simple verificación; luego admitiendo (1.11) y (1.12) como hipótesis inductiva,

$$\begin{aligned} P_{k+2}Q_{k+1} - Q_{k+2}P_{k+1} &= (x_{k+2}P_{k+1} + P_k)Q_{k+1} - (x_{k+2}Q_{k+1} + Q_k)P_{k+1} \\ &= P_kQ_{k+1} - Q_kP_{k+1} = -(-1)^k = (-1)^{k+1} \end{aligned}$$

y

$$\begin{aligned} P_{k+2}Q_k - Q_{k+2}P_k &= (x_{k+2}P_{k+1} + P_k)Q_k - (x_{k+2}Q_{k+1} + Q_k)P_k \\ &= x_{k+2}(P_{k+1}Q_k - Q_{k+1}P_k) = (-1)^k x_{k+2}. \end{aligned}$$

Teorema 1.12.1 Si $\langle a_0; a_1, \ldots, a_n \rangle$ es una f.c.f. entonces para todo $k \geq 0$,

$$[a_0; a_1, \ldots, a_k] = \frac{P_k(a_0, \ldots, a_k)}{Q_k(a_0, \ldots, a_k)}. \tag{1.13}$$

Prueba. En efecto,

$$[a_0; a_1] = a_0 + \frac{1}{a_1} = \frac{a_0 a_1 + 1}{a_1} = \frac{P_1(a_0, a1)}{Q_1(a_0, a_1)}.$$

Supongamos (1.13) es cierto para k, luego

$$\begin{aligned}
[a_0; a_1, \ldots, a_k, a_{k+1}] &= [a_0; a_1, \ldots, a_k + \tfrac{1}{a_{k+1}}] \\
&= \frac{P_k(a_0, a_1, \ldots, a_k + \frac{1}{a_{k+1}})}{Q_k(a_0, a_1, \ldots, a_k + \frac{1}{a_{k+1}})} \\
&= \frac{(a_k + \frac{1}{a_{k+1}})P_{k-1}(a_0, \ldots, a_k) + P_{k-2}(a_0, \ldots, a_{k-1})}{(a_k + \frac{1}{a_{k+1}})Q_{k-1}(a_0, \ldots, a_k) + Q_{k-2}(a_0, \ldots, a_{k-1})} \\
&= \frac{P_k(a_0, \ldots, a_k) + \frac{1}{a_{k+1}}P_{k-1}(a_0, \ldots, a_{k-1})}{Q_k(a_0, \ldots, a_k) + \frac{1}{a_{k+1}}Q_{k-1}(a_0, \ldots, a_{k-1})} \\
&= \frac{P_{k+1}(a_0, \ldots, a_{k+1})}{Q_{k+1}(a_0, \ldots, a_{k+1})}.
\end{aligned}$$

Definición 1.12.2 Una *fracción continua infinita* (denotada f.c.i.) es una sucesión infinita

$$\langle a_0; a_1, a_2, \ldots, a_n, \ldots \rangle$$

de números reales, donde $a_0 \in \mathbb{R}$ es cualquiera, y $a_i > 0$ para todo $i \geq 1$.
La f.c.i $\langle a_0; a_1, \ldots a_n, \ldots \rangle$ es convergente si

$$\lim_{n\to\infty} [a_0; a_1, \ldots, a_n]$$

existe, el cual se denota por $[a_0; a_1, \ldots, a_n, \ldots]$ y se dice que es el *valor* de la f.c.i.

Hacemos

$$p_n = P_n(a_0; a_1, \ldots, a_n), \qquad q_n = Q_n(a_0; a_1, \ldots, a_n).$$

Sea $r_n = \dfrac{p_n}{q_n} = [a_0; a_1, \ldots, a_n]$, entonces

$$r_{n+1} - r_n = \frac{(-1)^n}{q_{n+1}q_n}, \qquad r_{n+2} - r_n = \frac{(-1)^n a_{n+2}}{q_{n+2}q_n};$$

entonces se puede verificar fácilmente (utilizando inducción) que

$$r_0 < r_2 < \cdots < r_{2n} < r_{2n+1} < \cdots < r_3 < r_1.$$

Luego existen los límites $\lim\limits_{n\to\infty} r_{2n}$ y $\lim\limits_{n\to\infty} r_{2n+1}$ (pues ambas son sucesiones monótonas acotadas) y además $\lim\limits_{n\to\infty} r_{2n} \leq \lim\limits_{n\to\infty} r_{2n+1}$, de donde existe el límite $\lim\limits_{n\to\infty} r_n$ si y sólo si $\lim\limits_{n\to\infty} r_{2n} = \lim\limits_{n\to\infty} r_{2n+1}$, lo cual ocurre si y sólo si $\lim\limits_{n\to\infty} q_n q_{n+1} = \infty$.

Ahora podemos establecer el siguiente criterio de convergencia de una f.c.i.:

Teorema 1.12.2 La f.c.i. $\langle a_0; a_1, \ldots, a_n, \ldots \rangle$ es convergente si y sólo si $\sum_{n=1}^{\infty} a_j = \infty$.

Prueba. Supongamos que $\sum_{n=1}^{\infty} a_n < \infty$. Se cumple

$$q_n = a_n q_{n-1} + q_{n-2},$$

luego $q_n > q_{n-2}$, lo cual implica

$$q_n > q_{n-1} \quad \text{o} \quad q_{n-1} > q_{n-2},$$

ya que si $q_n \leq q_{n-1}$ y $q_{n-1} \leq q_{n-2}$, entonces $q_n \leq q_{n-2}$, una contradicción.
Como $\sum_{j=1}^{\infty} a_n$ converge, existe $N \in \mathbb{N}$ tal que $\sum_{j=N}^{\infty} a_j < 1$. Luego para todo $n \geq N$, se tiene $a_n < 1$; ahora si $n \geq N$ y $q_n > q_{n-1}$ entonces $q_n < a_n q_n + q_{n-2}$, esto es $q_n < \dfrac{q_{n-2}}{1 - a_n}$.

Si $n \geq N$ y $q_{n-1} > q_{n-2}$, entonces $q_n < (1 + a_n) q_{n-1} < \dfrac{q_{n-1}}{1 - a_n}$, luego para todo $n \geq N$, existe $l < n$ ($l = n-1$ ó $l = n-2$) tal que $q_n < \dfrac{q_l}{1 - a_n}$. Aplicando este proceso repetidas veces, obtenemos

$$q_n < \frac{q_s}{(1 - a_n)(1 - a_l) \cdots (1 - a_r)},$$

donde $n > l > \cdots > r \geq N > s$. Sea $q = \max_{0 \leq j < N} q_j$ y $\lambda = \prod_{j=N}^{\infty} (1 - a_j)$ (el producto infinito converge pues $\sum_{j=1}^{\infty} a_j$ converge), se tiene que $q_n < \dfrac{q}{\lambda}$ para todo $n \geq N$ de donde, si $\lambda > 0$, $q_{n+1} q_n < \dfrac{q^2}{\lambda^2}$. Luego sería imposible obtener $q_{n+1} q_n \to \infty$. Por tanto la f.c.i. diverge.

Probemos ahora que $\lambda > 0$. Tenemos

$$(1 - a_N)(1 - a_{N+1}) \geq 1 - a_N - a_{N+1},$$

además

$$(1 - a_N)(1 - a_{N+1})(1 - a_{N+2}) \geq (1 - a_N - a_{N+1})(1 - a_{N+2}) \geq 1 - a_N - a_{N+1} - a_{N+2}$$

y en general

$$(1 - a_N)(1 - a_{N+1})(1 - a_{N+2}) \cdots (1 - a_{N+k}) \geq 1 - a_N - a_{N+1} - \cdots - a_{N+k},$$

de donde

$$\lambda = \prod_{j=N}^{\infty} (1 - a_j) \geq 1 - \sum_{j=N}^{\infty} a_j > 0.$$

Ahora supongamos que $\sum_{j=1}^{\infty} a_j$ diverge. De a recurrencia $q_n = a_n q_{n-1} + q_{n-2}$ tenemos que $q_n > q_{n-2}$. Si $c = \min\{q_0, q_1\}$, entonces $q_n \geq c$ y $q_n \geq ca_n + q_{n-2}$ para todo $n \geq 0$. Aplicando repetidamente,

$$q_{2n} \geq q_0 + c\sum_{j=1}^{n} a_{2j}, \qquad q_{2n+1} \geq q_1 + c\sum_{j=1}^{n} a_{2j+1}$$

y sumando convenientemente

$$q_{2n} + q_{2n+1} \geq q_0 + q_1 + c\sum_{j=1}^{2n+1} a_j, \qquad q_{2n+2} + q_{2n+1} \geq q_0 + q_1 + c\sum_{j=1}^{2n+2} a_j.$$

De las últimas dos expresiones, tenemos que $q_n + q_{n-1} \geq q_0 + q_1 + c\sum_{j=1}^{n} a_j$. Ahora bien, sabemos que $q_n \geq c$, para todo n; y como $q_n + q_{n-1} > c\sum_{j=1}^{n} a_j$ entonces $q_n > \frac{c}{2}\sum_{j=1}^{n} a_j$ o $q_{n-1} > \frac{c}{2}\sum_{j=1}^{n} a_j$, de donde $q_n q_{n-1} > \frac{c^2}{2}\sum_{j=1}^{n} a_j \to \infty$ cuando $n \to \infty$. Por lo tanto, la f.c.i es convergente.

Lema 1.12.3 Si $\langle a_0; a_1, \ldots, a_n, \ldots \rangle$ es una f.c.i. convergente, entonces

(i) $[a_0; a_1, \ldots, a_n, \ldots] = a_0 + [0; a_1, \ldots, a_n, \ldots]$.

(ii) $[0; a_1, \ldots, a_n, \ldots] = \dfrac{1}{[a_1; a_2, \ldots, a_n, \ldots]}$.

(iii) $[a_0; a_1, \ldots, a_n, \ldots] = a_0 + \dfrac{1}{[a_1; a_2, \ldots, a_n, \ldots]}$.

(iv) $[a_0; a_1, \ldots, a_n, \ldots] = [a_0; a_1, \ldots, a_{k-1}, [a_k; a_{k+1}, \ldots, a_n, \ldots]]$.

(v) Si $a_i \geq 1$, para todo $i \geq 1$ entonces $0 < [0; a_1, \ldots, a_n, \ldots] \leq 1$, y

$$[0; a_1, \ldots, a_n, \ldots] = 1 \quad \text{si y sólo si} \quad \langle 0; a_1, \ldots, a_n, \ldots \rangle \text{ es igual a } \langle 0; 1 \rangle.$$

Si $n \geq 3$,

$$[0; a_1, \ldots, a_n] = \frac{1}{a_1 + \dfrac{1}{[a_2; a_3, \ldots, a_n]}} = \frac{1}{a_1 + \dfrac{1}{a_2 + [0; a_3, \ldots, a_n]}} \leq \frac{1}{a_1 + \dfrac{1}{a_2 + 1}} < 1.$$

Observe que (1.11) implica que $(p_n, q_n) = 1$.

Anteriormente vimos que

$$\frac{p_{2n}}{q_{2n}} \leq \frac{p_{2n+2}}{q_{2n+2}} \leq \alpha \leq \frac{p_{2n+1}}{q_{2n+1}} \leq \frac{p_{2n-1}}{q_{2n-1}}, \tag{1.14}$$

donde $\alpha = \lim_{n\to\infty} \frac{p_n}{q_n}$, el valor de la f.c.i $\langle a_0; a_1, \ldots, a_n, \ldots \rangle$. Ahora bien, si $k \geq 2$ e $i \geq 0$, calculemos

$$\frac{p_{k-1}(i+1)+p_{k-2}}{q_{k-1}(i+1)+q_{k-2}} - \frac{p_{k-1}i+p_{k-2}}{q_{k-1}i+q_{k-2}} = \frac{(-1)^k}{[q_{k-1}(i+1)+q_{k-2}][q_{k-1}i+q_{k-2}]}.$$

Las fracciones

$$\frac{p_{k-2}}{q_{k-2}}, \frac{p_{k-2}+p_{k-1}}{q_{k-2}+q_{k-1}}, \frac{p_{k-2}+2p_{k-1}}{q_{k-2}+2q_{k-1}}, \ldots, \frac{p_{k-2}+a_kp_{k-1}}{q_{k-2}+a_kq_{k-1}} = \frac{p_k}{q_k}$$

se ordenan en orden creciente si k es par, y decreciente si k es impar. Esto lo justifica la siguiente proposición, cuya prueba es trivial:

Proposición 1.12.3 Si $a, b, c, d \in \mathbb{Z}$, con $b, d \in \mathbb{N}$, $(a, b) = 1 = (c, d)$, entonces la *mediana* entre las dos fracciones $\frac{a}{b} \leq \frac{c}{d}$ es $\frac{a+c}{b+d}$ y cumple

$$\frac{a}{b} \leq \frac{a+c}{b+d} \leq \frac{c}{d}.$$

Lema 1.12.4 Si α es el valor de la f.c.i. $\langle a_0; a_1, \ldots, a_n, \ldots \rangle$ entonces

$$\frac{1}{2q_nq_{n+1}} < \left|\alpha - \frac{p_n}{q_n}\right| < \frac{1}{q_nq_{n+1}} < \frac{1}{q_n^2}.$$

Prueba. Dividimos el estudio en dos casos:
Caso $n = 2k$. En este caso

$$\frac{p_{2k}}{q_{2k}} \leq \frac{p_{2k+1}+p_{2k}}{q_{2k+1}+q_{2k}} \leq \alpha \leq \frac{p_{2k+1}}{q_{2k+1}},$$

luego

$$\alpha - \frac{p_{2k}}{q_{2k}} \geq \frac{p_{2k+1}+p_{2k}}{q_{2k+1}+q_{2k}} - \frac{p_{2k}}{q_{2k}} = \frac{1}{q_{2k}(q_{2k+1}+q_{2k})} > \frac{1}{2q_{2k}q_{2k+1}},$$

pues $q_{2k+1} > q_{2k}$ si $a_i \in \mathbb{N}$, para $i \geq 1$.
Caso $n = 2k+1$. Se cumple

$$\frac{p_{2k+2}}{q_{2k+2}} \leq \alpha \leq \frac{p_{2k+2}+p_{2k+1}}{q_{2k+2}+q_{2k+1}} \leq \frac{p_{2k+1}}{q_{2k+1}},$$

luego

$$\frac{p_{2k+1}}{q_{2k+1}} - \alpha \geq \frac{1}{q_{2k+1}(q_{2k+1}+q_{2k+2})} > \frac{1}{2q_{2k+1}q_{2k+2}}.$$

Finalmente, de (1.14) tenemos para n par ó impar se tiene

$$\left|\alpha - \frac{p_n}{q_n}\right| < \left|\frac{p_{n+1}}{q_{n+1}} - \frac{p_n}{q_n}\right| = \frac{|p_{n+1}q_n - p_nq_{n+1}|}{q_{n+1}q_n} = \frac{1}{q_{n+1}q_n}.$$

Teorema 1.12.4

(i) Todo número racional es el valor de dos fracciones continuas, ambas finitas y de las formas

$$\langle a_0; a_1, \ldots, a_{N+1}\rangle, \qquad \langle a_0; a_1, \ldots, a_{N+1} - 1, 1\rangle,$$

donde $a_{N+1} \neq 1$.

(ii) Todo número irracional es el valor de exactamente una fracción continua convergente.

(iii) Si $a \in \mathbb{R}$, definimos $\alpha_0 = a$,

$$\alpha_{k+1} = \frac{1}{\alpha_k - [\alpha_k]}, \quad a_k = [\alpha_k], \qquad \text{para todo } k \geq 0,$$

entonces

$$\lim_{n\to\infty} [a_0; a_1, \ldots, a_n] = a.$$

Prueba. Si $a \in \mathbb{Z}$ tomamos $\langle a\rangle$. Si $a \in \mathbb{Q}$ vamos a probar que usando el procedimiento (iii) se obtiene una f.c.f.
Por definición, $a \in \mathbb{Q}$ implica $\alpha_n \in \mathbb{Q}$. Ahora $a = [a_0; \alpha_1]$ y en general,

$$a = [a_0; a_1, \ldots, a_{n-1}, \alpha_n].$$

Si $\alpha_n = \dfrac{A}{B}$, con $A, B \in \mathbb{Z}$, $B > 1$ y $(A, B) = 1$ (fracción reducida), entonces

$$\alpha_n - a_n = \frac{A - Ba_n}{B} = \frac{C}{B} < 1$$

implica $C < B$; además $\alpha_{n+1} = \dfrac{B}{C}$. De esta manera, en cada paso del algoritmo se reduce el denominador de α_n; terminamos cuando $\alpha_N \in \mathbb{N}$ y $\alpha_N > 1$.
Si $a \notin \mathbb{Q}$ entonces $\alpha_n \notin \mathbb{Q}$, para todo $n \geq 0$ y el algoritmo (iii) nos da una f.c.i $\langle a_0; a_1, \ldots, a_n, \ldots\rangle$.
Tenemos

$$\begin{aligned} a &= [a_0; a_1, \ldots, \alpha_n] = \frac{P_n(a_0; a_1, \ldots, \alpha_n)}{Q_n(a_0; a_1, \ldots, \alpha_n)} \\ &= \frac{\alpha_n P_{n-1}(a_0; a_1, \ldots, a_{n-1}) + P_{n-2}(a_0; a_1, \ldots, a_{n-2})}{\alpha_n Q_{n-1}(a_0; a_1, \ldots, a_{n-1}) + Q_{n-2}(a_0; a_1, \ldots, a_{n-2})} \end{aligned}$$

de donde

$$a - \frac{p_n}{q_n} = \frac{(p_{n-1}q_{n-2} - q_{n-1}p_{n-2})(\alpha_n - a_n)}{(q_{n-1}a_n + q_{n-2})(q_{n-1}\alpha_{n-1} + q_{n-2})},$$

y observando que $a_n \leq \alpha_n < a_n + 1$, obtenemos

$$\left| a - \frac{p_n}{q_n} \right| < \frac{1}{(q_{n-1}a_n + q_{n-2})^2} = \frac{1}{q_n^2}.$$

De esto último, y como $q_n \to \infty$ cuando $n \to \infty$, entonces

$$\left| a - \frac{p_n}{q_n} \right| < \frac{1}{q_n^2} \to 0$$

cuando $n \to \infty$. Esto prueba la convergencia de $r_n = p_n/q_n$.
Ahora veamos la unicidad. Si $\langle a_0; a_1, \ldots, a_n, \ldots \rangle$ y $\langle a_0'; a_1', \ldots, a_n', \ldots \rangle$ son dos sucesiones tales que

$$[a_0; a_1, \ldots, a_n, \ldots] = [a_0'; a_1', \ldots, a_n', \ldots]$$

vamos a probar que $a_i = a_i'$, para todo $i \geq 0$ bajo ciertas condiciones.
Supongamos que $[a_0; a_1, \ldots, a_n, \ldots] < 1$ y $[a_0'; a_1', \ldots, a_n', \ldots] < 1$, entonces

$$a_0 + [0; a_1, \ldots, a_n, \ldots] = a_0' + [0; a_1', \ldots, a_n', \ldots]$$

y luego

$$|a_0 - a_0'| = |[0; a_1, \ldots, a_n, \ldots] - [0; a_1', \ldots, a_n', \ldots]|;$$

el primer miembro de esta última igualdad es un entero y el segundo se encuentra en el intervalo $[0, 1\rangle$, y por lo tanto ambos miembros son iguales a cero, esto es $a_0 = a_0'$.
Ahora de la igualdad

$$[a_1; a_2, \ldots, a_n, \ldots] = [a_1'; a_2', \ldots, a_n', \ldots]$$

y si $[a_1; a_2, \ldots, a_n, \ldots] < 1$ y $[a_1'; a_2', \ldots, a_n', \ldots] < 1$, como en el caso anterior obtenemos $a_1 = a_1'$. Repitiendo el proceso anterior, obtenemos

$$a_i = a_i', \text{ para todo } i \geq 0.$$

Observe que para obtener la unicidad, no podemos admitir $a_{n+1} = 1$ como último término la fracción continua (caso racional).

1.13 Sobre la densidad de los números primos

Definamos, para $x \geq 2$ entero, la función

$$\pi(x) = \#\{p \in \mathbb{N} : 2 \leq p \leq x, \, p \text{ primo}\}.$$

Uno de los más importantes problemas de la matemáticas en el pasado era el de determinar el comportamiento asintótico de $\pi(x)$. El teorema del número primo, demostrado en 1896 por de la Vallée Poussin y Hadamard (independientemente) establece que

$$\lim_{x \to \infty} \frac{\pi(x) \log x}{x} = 1,$$

de donde $\pi(x)$ se comporta asintóticamente como $x/\log x$. Este teorema asimismo mide la *densidad* de los números primos dentro de los números naturales.

Para nuestros propósitos sólo requeriremos una versión débil del teorema del número primo. Antes de ello establecemos un lema.

Lema 1.13.1 Para $n \in \mathbb{N}$, sea $w_p(n)$ el exponente de p en la factorización de $n!$; entonces

$$w_p(n) = \sum_{k=1}^{\infty} \left\lfloor \frac{n}{p^k} \right\rfloor .$$

La suma es finita ya que para $k > \log_p n$, se tiene que $\lfloor n/p^k \rfloor = 0$.

Prueba. Haremos la prueba por inducción sobre n. Para $n = 1$ el resultado es trivial, $w_p(1) = 0$. Notemos que

$$\begin{aligned} \left\lfloor \frac{n+1}{p^j} \right\rfloor - \left\lfloor \frac{n}{p^j} \right\rfloor &= \sum_{k \le \frac{n+1}{p^j}} 1 - \sum_{k \le \frac{n}{p^j}} 1 = \sum_{\frac{n}{p^j} < k \le \frac{n+1}{p^j}} 1 \\ &= \sum_{k, n < kp^j \le n+1} 1 = \begin{cases} 1, & \text{si existe } k : kp^j = n+1, \\ 0, & \text{en otro caso,} \end{cases} \end{aligned}$$

o equivalentemente,

$$\left\lfloor \frac{n+1}{p^j} \right\rfloor - \left\lfloor \frac{n}{p^j} \right\rfloor = \begin{cases} 1, & \text{si } p^j | n+1, \\ 0, & \text{si } p^j \not| n+1. \end{cases}$$

De ahí es claro que

$$w_p(n+1) - w_p(n) = \sum_{k=1}^{\infty} \left(\left\lfloor \frac{n+1}{p^j} \right\rfloor - \left\lfloor \frac{n}{p^j} \right\rfloor \right) = k,$$

es el exponente de p en la factorización de $n+1$. Por hipótesis inductiva, $w_p(n)$ es el exponente de p en la factorización de $n!$, de donde

$$w_p(n+1) = w_p(n) + k$$

es el exponente de p en la factorización de $n!(n+1) = (n+1)!$.

Teorema 1.13.1 (Tchebycheff) Si $x \ge 2$, entonces

$$\frac{x}{\log_2 x} < \pi(x) < \frac{5x}{\log_2 x}.$$

Prueba. Observemos primero que $\binom{2n}{n} = \frac{(2n)!}{n!^2}$ es múltiplo de todos los primos p tales que $n < p \leq 2n$. Como

$$\binom{2n}{n} < \sum_{0 \leq k \leq 2n} \binom{2n}{n} = 2^{2n},$$

entonces

$$\prod_{\substack{p \text{ primo} \\ n < p \leq 2n}} p < 2^{2n}.$$

Como hay $\pi(2n) - \pi(n)$ de estos primos, entonces

$$n^{\pi(2n)-\pi(n)} < \prod_{\substack{p \text{ primo} \\ n < p \leq 2n}} p < 2^{2n},$$

de donde $(\pi(2n) - \pi(n)) \log_2 n < 2n$ y

$$\pi(2n) - \pi(n) < \frac{2n}{\log_2 n}.$$

Por lo tanto para $k \geq 1$ se tiene que

$$\pi(2^{k+1}) \leq \frac{5 \cdot 2^k}{k}$$

(para $k \leq 5$ se sigue del hecho que $\pi(n) \leq n/2$, a partir de $k = 5$ utilizamos inducción sobre k y la propiedad demostrada). Como además la función $f(x) = x/\log_2 x$ es creciente para $x \geq 3$, entonces si $2^k < x \leq 2^{k+1}$, tenemos que

$$\pi(x) \leq \frac{5 \cdot 2^k}{k} \leq \frac{5x}{\log_2 x}.$$

Probemos la otra desigualdad. Por el lema anterior, el exponente de p en $\binom{2n}{n} = \frac{(2n)!}{n!^2}$ es

$$\alpha_p = \sum_{k=1}^{\infty} \left(\left\lfloor \frac{2n}{p^k} \right\rfloor - 2 \left\lfloor \frac{n}{p^k} \right\rfloor \right).$$

Observemos que $\lfloor 2x \rfloor - 2\lfloor x \rfloor$ sólo toma los valores 0 y 1: En efecto, como $\lfloor 2x \rfloor \leq 2x$ y el lado izquierdo es un entero, entonces

$$\lfloor 2x \rfloor \leq \lfloor 2x \rfloor$$

y como

$$\lfloor 2x \rfloor - 2\lfloor x \rfloor \leq 2x - 2\lfloor x \rfloor = 2(x - \lfloor x \rfloor) < 2,$$

y nuevamente el lado izquierdo es entero, entonces $\lfloor 2x \rfloor - 2\lfloor x \rfloor \leq 1$, y de donde

$$0 \leq \lfloor 2x \rfloor - 2\lfloor x \rfloor \leq 1.$$

Como para $k > \log_2(2n)/\log_2 p$ se cumple que $\lfloor 2n/p^k \rfloor - 2\lfloor n/p^k \rfloor$ se anula, y este término toma a lo más el valor 1, entonces $\alpha_p \leq \log_2(2n)/\log_2 p$. Además, si $n < p \leq 2n$ entonces $\alpha_p = 1$. Escribiendo

$$\binom{2n}{n} = \prod_{p<2n} p^{\alpha_p},$$

la factorización en primos de $\binom{2n}{n}$, entonces

$$\begin{aligned} \log_2 \binom{2n}{n} &= \sum_{p<2n} \alpha_p \log_2 p = \sum_{p\leq n} \alpha_p \log_2 p + \sum_{n<p\leq 2n} \alpha_p \log_2 p \\ &\leq \pi(n)\log_2(2n) + (\pi(2n) - \pi(n))\log_2(2n) = \pi(2n)\log_2(2n), \end{aligned}$$

de esto y de

$$\binom{2n}{n} = \frac{2n}{n} \cdot \frac{2n-1}{n-1} \cdot \cdots \cdot \frac{n+1}{1} \geq 2^n,$$

tenemos

$$\pi(2n) \geq \log_2 \binom{2n}{n} / \log_2(2n) \geq \frac{n}{\log_2(2n)}.$$

Esto prueba que

$$\pi(x) \geq \frac{x}{\log_2 x}$$

para todo x par; es obvio entonces para x impar ya que $\pi(2k-1) = \pi(2k)$ para todo k entero.

1.14 Números aleatorios

En los siguientes capítulos estudiaremos algunos algoritmos que requieren una muestra de datos "aleatoria" durante su ejecución; estos algoritmos son llamados *algoritmos probabilísticos.* En algunos casos, la veracidad de la salida de estos algoritmos probabilísticos depende de hechos de la teoría de probabilidades; estos últimos algoritmos son llamados *aleatorizados.* Aunque no siempre dan con el resultado correcto (de hecho las salidas son "probablemente ciertas"), su velocidad de ejecución es por mucho superior a la de los algoritmos *determinísticos* (algoritmos que no requieren datos aleatorios para su ejecución).

Para nuestros propósitos bastará con proporcionar una muestra de datos *uniformemente distribuida*, esto es, una entrada en la cual cada dato posible es igualmente probable; y no necesariamente datos completamente aleatorios.

El problema que se presenta es el de escoger números enteros en un intervalo dado $[a, b]$. Podemos escoger de hecho el intervalo $[0, m-1]$ sin pérdida de generalidad. Consideremos la siguiente sucesión de números enteros:

$$x_{n+1} = (ax_n + c) \bmod m, \qquad n \geq 0; \tag{1.15}$$

donde m, a, c, x_0 son enteros tales que

$$0 \leq m, a, c, x_0 < m.$$

Esta sucesión es conocida como *sucesión congruencial lineal.* Los números m, a, c y x_0 son llamados *módulo*, *multiplicador*, *incremento* y *valor inicial* (o *semilla*) respectivamente.

Observamos que el generador de números presentado obedece a la forma

$$x_{n+1} = f(x_n), \tag{1.16}$$

donde $f : E \to E$ y E conjunto finito con m elementos. Como E es finito, existirán $r < s$ tales que $f(x_r) = f(x_s)$; escribiendo $k = s - r$ (es obvio que $k \leq m$), tenemos que $f(x_r) = f(x_{r+k})$ y aplicando la recurrencia (1.16) repetidamente, concluimos que la sucesión tiene *período* k, esto es

$$f(x_n) = f(x_{n+k})$$

para todo $n \geq r$. A causa de esto, la sucesión dada no será aleatoria. Sin embargo, ese no será un problema ya que para nuestros propósitos será suficiente obtener una sucesión uniformemente distribuida de datos; buscaremos entonces el período máximo para nuestra sucesión, en el caso de la sucesión (1.16) buscaremos el período m.

El siguiente teorema nos dice cuándo el período máximo es alcanzado.

Teorema 1.14.1 La sucesión congruencial lineal definida por m, a, c y x_0 posee período m si y sólo si

(i) $(c, m) = 1$;

(ii) $b = a - 1$ es múltiplo de p, para cada primo p que divide a m;

(iii) b es un múltiplo de 4, si m es múltiplo de 4.

No damos la prueba de este teorema ya que no corresponde a este trabajo el estudiar la generación de números aleatorios. Sin embargo necesitamos medir el tiempo necesario para generar números aleatorios; por ello incluimos esta sección. De la ecuación (1.15) es claro que calcular x_{n+1} a partir de x_n tiene complejidad $O(\log^2 m)$ (que es el costo de la división entre m, la operación más costosa).

Existen otros tipos de generadores de gran importancia, como los generadores de *bits* (ceros y unos), muy convenientes para el trabajo con computadoras. Consideremos la sucesión definida por

$$x_n = (x_{n-24} + x_{n-55}) \bmod m, \qquad n \geq 55,$$

donde m es par, y los valores $x_0, \ldots, x_{54}$ son enteros arbitrarios no todos pares. Entonces la sucesión de bits definida por

$$b_n = x_n \bmod 2$$

tiene período $2^{55} - 1$. Esto implica que la sucesión x_n tiene un período mayor o igual a $2^{55} - 1$. De hecho, se sabe que si $m = 2^e$, entonces el periodo de la sucesión x_n es $2^{e-1}(2^{55} - 1)$. Esto no es extraño si nos percatamos que la sucesión x_n es un caso especial de una sucesión de la forma

$$x_n = f(x_{n-1}, \ldots, x_{n-k}), \qquad 0 \leq x_n < m,$$

donde es fácil ver que el máximo periodo concebible es m^k; puede probarse que para valores cualquiera de m, k existe una función f tal que la sucesión arriba tiene periodo precisamente igual a m^k.

Para mayor información sobre la generación de números aleatorios, consulte [2].

Capítulo 2

Detectando números compuestos

El primer problema que atacaremos es el de detectar si un número entero dado es compuesto. Como veremos, los problemas de determinar primalidad y determinar si un número es compuesto son problemas distintos. Utilizando los métodos estudiados para obtener números "aleatorios" de la sección 1.14, construimos algoritmos veloces que justifican su funcionamiento gracias a ciertos hechos básicos de probabilidades; estos algoritmos son conocidos como *algoritmos probabilísticos*. Todos los algoritmos presentados en este capítulo, con excepción del primero, son probabilísticos; además poseen la siguiente particularidad: uno de estos algoritmos detecta efectivamente números compuestos, pero no puede probar que un número es primo, un número que supera estos tests es *probablemente primo* (un entero con una probabilidad bastante alta de ser primo), aunque no un primo certificado. Es ahí donde se requieren los algoritmos de los siguientes capítulos.

2.1 Un primer intento

Si un número es compuesto, entonces podemos escribir n como $n = kl$, donde $1 < k < n$. Además uno de los factores k, l debe ser $\leq \sqrt{n}$. Esto nos da una primera idea de cómo verificar que un número es compuesto: hallar un factor no trivial.

Algoritmo 2.1.1 Este algoritmo intenta hallar un factor no trivial de un entero n; de no existir dicho factor, el número es primo.

Entrada: entero n
$k = 2$
mientras $(n \not\equiv 0 (\bmod k) \wedge k \leq \sqrt{n})$
$\quad k = k + 1$
Salida: si $k \leq \sqrt{n}$ "k divide a n"
sino "n es primo"

Note que el costo de este algoritmo es $O(\sqrt{n}\log^2 n)$ ($\sqrt{n}$ iteraciones de costo $O(\log^2 n)$, que es el costo de la división). Esto muestra que el algoritmo es exponencial, lo que lo hace inútil para estudiar números relativamente grandes. Como veremos a continuación, podemos probar que un número es compuesto sin necesidad de hallar alguno de sus factores.

2.2 Números de Carmichael

Vamos a usar el pequeño teorema de Fermat para elaborar un algoritmo probabilístico para determinar primalidad.

Teorema 2.2.1 Si $a^{n-1} \not\equiv 1 (\bmod n)$ para algún $a \in \mathbb{Z}_n^*$, esto ocurre para por lo menos el 50% de los elementos del $\mathbb{Z}_n^*$.

Prueba. Sean $a_1, \ldots, a_t$ todos los elementos $a_i \in \mathbb{Z}_n^*$ tales que

$$a_i^s \equiv 1 (\bmod n).$$

Observe que $aa_1, \ldots, aa_t$ son elementos distintos de $\mathbb{Z}_n^*$ y que

$$(aa_i)^s \equiv a^s a_i^s \equiv a^s \not\equiv 1 (\bmod n)$$

(y los aa_i son distintos a los a_j). Por lo tanto se tiene el resultado. ■

Algoritmo 2.2.1 Este algoritmo intenta determinar si un número entero dado es compuesto.

Entrada: n entero
k parámetro de seguridad

$i = 1$

repetir
  escogemos aleatoriamente
  a con $1 < a < n$
  si $(a, n) = 1 \wedge a^{n-1} \equiv 1 (\bmod n)$
    $i = i + 1$
  sino
    $i = 0$
mientras $(i > 0 \wedge i \leq k)$

Salida: si $i = 0$ "compuesto"
sino "probablemente primo"

El costo del algoritmo es $O(k \log^3 n)$, y si k es fijo el algoritmo es polinomial.
Si n pasa el test con k números enteros elegidos aleatoriamente, entonces la probabilidad de que exista $a \in \mathbb{Z}_n^*$ con

$$a^{n-1} \not\equiv 1 (\mathrm{mod}\, n),$$

es menor que $(\frac{1}{2})^k$. En particular, si $k = 200$, la probabilidad es $2^{-200} < 10^{-60}$, ¡casi nada! Sin embargo, nos encontramos con un problema: no demostramos que si n es compuesto, entonces existe $a \in \mathbb{Z}_n^*$ tal que $a^{n-1} \not\equiv 1 (\mathrm{mod}\, n)$. ¿Será que existe un número compuesto n tal que $a^{n-1} \equiv 1 (\mathrm{mod}\, n)$ para todo $a \in \mathbb{Z}_n^*$? La respuesta es sí, lo que nos lleva a la siguiente

Definición 2.2.1 Un número compuesto n es llamado *número de Carmichael* si y sólo si

$$a^{n-1} \equiv 1 (\mathrm{mod}\, n), \quad \text{para todo } a \in \mathbb{Z}_n^*.$$

Ejemplo 2.2.1 El menor número de Carmichael es $n = 561$.

Aún siendo 561 un número pequeño, se hace difícil comprobar que es de Carmichael por definición. Felizmente tenemos la siguiente caracterización:

Teorema 2.2.2 (Korselt) Un número compuesto n es de Carmichael si y sólo si todo factor primo p de n satisface:

(a) $p^2 \not| n$,

(b) $p - 1 | n - 1$.

Prueba. Supongamos que n es de Carmichael, y sea p un primo, $p|n$. Supongamos ahora que $p^2|n$. Sea g una raíz primitiva módulo p^2, y $n' = \prod_{\substack{q \neq p \\ q \text{ primo}}} q$. Por el teorema chino del resto, existe un entero b tal que

$$b \equiv g (\mathrm{mod}\, p^2),$$
$$b \equiv 1 (\mathrm{mod}\, n),$$

luego b es una raíz primitiva módulo p^2 (por g), y además $(b, n) = 1$ por construcción. Probemos que $b^{n-1} \not\equiv 1 (\mathrm{mod}\, n)$. Por reducción al absurdo, supongamos que

$$b^{n-1} \equiv 1 (\mathrm{mod}\, n).$$

Como $p^2|n$, $b^{n-1} \equiv 1 (\mathrm{mod}\, p^2)$. Pero como $\exp_{p^2}(b) = \exp_{p^2}(g) = p(p-1)$, entonces $p(p-1)|(n-1)$. Sin embargo

$$n - 1 = -1 (\mathrm{mod}\, p),$$

pues $p|n$, de donde $p(p-1)\not| n-1$, una contradicción.
Ahora probemos que $p-1|n-1$. En efecto, tomando una raíz primitiva módulo p,

$$g^{n-1} \equiv 1 (\bmod n),$$

de donde

$$g^{n-1} \equiv 1 (\bmod p)$$

y por tanto $\exp_p(g) = p-1|n-1$.
Recíprocamente, si (a) y (b) se satisfacen para todo factor primo p de n, entonces $n = p_1 \cdots p_r$ con p_i primos distintos (n es libre de cuadrados por (a)). Para cada p_i, por (b) existe k entero con

$$n-1 = k(p_i - 1)$$

y

$$b^{n-1} \equiv (b^{p_i-1})^k \equiv 1^k \equiv 1 (\bmod p_i)$$

por el pequeño teorema de Fermat ya que $((b, p_i) = 1)$. Como los p_i's son distintos,

$$b^{n-1} \equiv 1 (\bmod p_1 \cdots p_r).$$

Teorema 2.2.3 Un número de Carmichael es producto de al menos tres primos distintos.

Prueba. Basta probar que no podemos escribir un número de Carmichael n como $n = pq$, con $p \neq q$ por el teorema anterior. Supongamos, por reducción al absurdo, que esto ocurra. Podemos suponer que $p < q$. De la identidad

$$p - 1 = (n-1) - p(q-1)$$

y como $q-1|n-1$, tenemos que $q-1|p-1$. Pero

$$0 < p-1 < q-1,$$

una contradicción.

Surgen entonces las preguntas naturales: ¿Existen infinitos números de Carmichael? La respuesta es que sí. El siguiente teorema muestra además por lo menos "cuántos" números de Carmichael existen (ver [14]):

Teorema 2.2.4 (Alford, Pomerance, Granville) Dado x, sea $C(x)$ la cantidad de números de Carmichael menores o iguales a x. Entonces existe $n_0 \in \mathbb{N}$ tal que, para $x > n_0$ se tiene que

$$C(x) > x^{2/7}.$$

Dado un número de Carmichael, el algoritmo 2.2.1 sólo fallará si encontramos a con $(a, n) \neq 1$. Como para cada n, pueden existir pocos de estos números a, debemos buscar un mejor algoritmo, uno que no requiera hipótesis adicionales sobre n.

2.3 Test de Miller-Rabin

Tratamos de mejorar el método anterior, evitando asumir hipótesis adicionales sobre un número compuesto para aumentar las probabilidades de éxito de nuestro algoritmo.
El test de Miller (1976) tiene su punto de partida en la siguiente observación:
Sea $n > 0$ primo impar y $1 < b < n-1$ entero. Como b es impar, escribimos

$$n - 1 = 2^s t,$$

donde t es un entero impar y $k \geq 1$.
Sea $A = \{0 \leq i \leq s : b^{2^i t} \equiv 1 (\mathrm{mod}\, n)\}$. Por el pequeño teorema de Fermat, $s \in A$ así que $A \neq \emptyset$.
Sea $j = \min A$. Supongamos que $j \geq 1$. Tenemos que $b^{2^j t} \equiv 1 (\mathrm{mod}\, n)$ y como n es primo

$$(b^{2^{j-1}t})^2 \equiv 1 (\mathrm{mod}\, n) \quad \text{implica} \quad b^{2^{j-1}t} \equiv \pm 1 (\mathrm{mod}\, n).$$

Pero no puede darse que $b^{2^{j-1}t} \equiv 1 (\mathrm{mod}\, n)$, pues de ser así $j - 1 \in A$, contradiciendo la minimalidad de j. Luego

$$b^{2^{j-1}t} \equiv -1 (\mathrm{mod}\, n).$$

Si $j = 0$, tenemos simplemente

$$b^t \equiv 1 (\mathrm{mod}\, n).$$

Esto motiva la siguiente

Definición 2.3.1 Decimos que un entero impar n es un *pseudoprimo en relación a la base b* cuando

$$b^t \equiv 1 (\mathrm{mod}\, n) \quad \text{o} \quad b^{2^i t} \equiv -1 (\mathrm{mod}\, n)$$

para algún $0 \leq i \leq s-1$, donde $n - 1 = 2^s t$ y t es impar.

El análisis anterior se resume de la siguiente manera:

Teorema 2.3.1 Si n es primo, $1 < b < n-1$ un entero, entonces n es un pseudoprimo en relación a la base b.

Es claro que el test de Miller-Rabin es una generalización del test presentado en la sección anterior; sin embargo, el test de Miller-Rabin es evidentemente superior como lo muestra el siguiente resultado.

Proposición 2.3.2 (Monier-Rabin) Si $n > 9$ es impar y compuesto, entonces n es pseudoprimo a lo más en relación al 25% de las bases a coprimas con n tales que $1 \leq a \leq n-1$.

Para la prueba de esta proposición necesitamos antes del siguiente

Lema 2.3.1 Sea n impar compuesto con $n-1=2^s t$, t impar. Sea $\nu(n)$ el mayor entero tal que $2^{\nu(n)}|p-1$, para todo primo p que divide a n. Si n es pseudoprimo respecto a la base a, entonces

$$a^{2^{\nu(n)-1}t} \equiv \pm 1 (\bmod n).$$

Demostración (del lema). Si $a^t \equiv 1 (\bmod n)$ la conclusión es evidente. Supongamos que $a^{2^i t} \equiv -1 (\bmod n)$ y sea p un factor primo de n, entonces

$$a^{2^i t} \equiv -1 (\bmod p). \tag{2.1}$$

Si $k = \min\{j \geq 0 : a^j \equiv 1 (\bmod p)\} = \exp_p(a)$, entonces $k|2^{i+1}t$ pero $k \not| 2^i t$ por (2.1), y por lo tanto $2^{i+1}|k$. Pero además $k|p-1$, entonces $2^{i+1}|p-1$. Como esto ocurre para todo p primo que divide a n, entonces $i+1 \leq \nu(n)$, y por lo tanto

$$a^{2^{\nu(n)-1}y} \equiv 1 (\bmod n) \text{ o } -1 (\bmod n)$$

en caso que $i+1 < \nu(n)$ o $i+1 = \nu(n)$.

Sea

$$\overline{\mathcal{S}}(n) = \{a (\bmod n) : a^{2^{\nu(n)-1}t} \equiv \pm 1 (\bmod n)\}, \quad \overline{S}(n) = \#\overline{\mathcal{S}}(n).$$

Lema 2.3.2 Sea $\omega(n)$ el número de factores primos distintos de n. Entonces

$$\overline{S}(n) = 2 \cdot 2^{(\nu(n)-1)\omega(n)} \prod_{p|n} (t, p-1).$$

Demostración. Sea $m = 2^{\nu(n)-1}t$. Supongamos que la factorización en primos de n es

$$n = p_1^{j_1} p_2^{j_2} \cdots p_k^{j_k},$$

donde $k = \omega(n)$. Tenemos que

$$a^m = 1 (\bmod n) \text{ si y sólo si } a^m = 1 (\bmod p_i^{j_i}), \quad i = 1, 2, \ldots, k.$$

Como existen raíces primitivas módulo $p_i^{j_i}$ para p_i primo impar y $j \in \mathbb{Z}$, $j > 0$, entonces el número de soluciones $a (\bmod p_i^{j_i})$ para la congruencia binomial $a^m \equiv 1 (\bmod p_i^{j_i})$ es $(m, \varphi(p_i^{j_i}))$ y

$$(m, \varphi(p_i^{j_i})) = (m, p_i^{j_i - 1}(p_i - 1)) = (m, p_i - 1) = 2^{\nu(n)-1}(t, p_i - 1)$$

(observe que $m|n-1$ implica que $p_i \not| m$). Por el teorema chino del resto, el número de soluciones $a(\bmod n)$ para $a^m \equiv 1(\bmod n)$ es

$$\prod_{i=1}^{k}(2^{\nu(n)-1}\cdot(t,p_i-1)) = 2^{(\nu(n)-1)\omega(n)}\prod_{p|n}(t,p-1).$$

Para completar la prueba, probaremos que hay exactamente el mismo número de soluciones para la congruencia $a^m \equiv -1(\bmod n)$. Note que $a^m \equiv -1(\bmod p_i^{j_i})$ si y sólo si

$$a^{2m} \equiv 1(\bmod p_i^{j_i}) \quad \text{y} \quad a^m \equiv 1(\bmod p_i^{j_i}).$$

Como $2^{\nu(n)}|p_i-1$ se sigue como antes que el número de soluciones de $a^m \equiv -1(\bmod p_i^{j_i})$ es

$$2^{\nu}(n)\cdot(t,p_i-1) - 2^{\nu(n)-1}(t,p_i-1) = 2^{\nu(n)-1}(t,p_i-1) = 2^{\nu(n)-1}(t,p_i-1).$$

De ahí se sigue el resultado.

Prueba de la proposición 2.3.2. Por el lema 2.3.1, basta probar que $\overline{S}(n)/\varphi(n) \leq 1/4$ cuando n es un número impar compuesto > 9. Del lema anterior

$$\frac{\varphi(n)}{\overline{S}(n)} = \frac{1}{2}\prod_{p^a\|n} p^{a-1}\frac{p-1}{2^{\nu(n)-1}(t,p-1)}.$$

Cada factor $(p-1)/(2^{\nu(n)-1}(t,p-1)))$ es un entero par, de modo que $\varphi(n)/\overline{S}(n)$ es un entero. Ahora, si $\omega(n) \geq 3$, entonces $\varphi(n)/\overline{S}(n) \geq 4$. Si $\omega(n) = 2$ y n no es libre de cuadrados, el producto de los p^{a-1} es al menos 3, luego $\varphi(n)/\overline{S}(n) \geq 6$.

Ahora supongamos que $n = pq$, donde $p < q$ son primos. Si $2^{\nu(n)+1}|q-1$, entonces $2^{\nu(n)-1}(t,q-1) \leq (q-1)/4$ y $\varphi(n)/\overline{S}(n) \geq 4$. Podemos suponer entonces que $2^{\nu(n)}\|q-1$. Note que

$$n-1 = pq-1 = p(q-1)+(p-1)$$

implica que $n-1 \equiv p-1(\bmod q-1)$, y luego $q-1 \not| n-1$. Esto implica que existe un primo impar dividiendo $q-1$ con una potencia mayor con la que divide $n-1$, esto es, $2^{\nu(n)-1}(t,q-1) \leq (q-1)/6$. Concluimos que $\varphi(n)/\overline{S}(n) \geq 6$ en este caso. Finalmente, supongamos que $\omega(n) = 1$, i.e., $n = p^a$, donde $a \geq 2$. Entonces $\varphi(n)/\overline{S}(n) = p^{a-1}$, de donde $\varphi(n)/\overline{S}(n) \geq 5$, excepto cuando $p^a = 9$.

Algoritmo 2.3.1 (Miller-Rabin) Este algoritmo determina que un número entero dado n es compuesto con certeza, hallando un divisor propio de n o una base a con respecto a la cual n no es pseudoprimo, o determina que no existe una base tal salvo por una probabilidad menor a $(1/4)^k$.

```
Entrada: n entero impar
         k parámetro de seguridad
i = 1
mientras (i > 0 ∧ i ≤ k)
    escoger aleatoriamente
            a con 2 < a < n − 2
    si (a, n) ≠ 1
        i = 0
    sino
        y = a^t mod n
        si y ≠ 1 ∧ y ≠ n − 1
            j = 1
            mientras (j > 0 ∧ j ≤ s − 1 ∧ y ≠ n − 1)
                y = y^2 mod n
                si y = 1
                    j = 0
                sino
                    j = j + 1
            si y ≠ n − 1
                i = 0
        si (i > 0)
            i = i + 1
Salida: si i = 0 "compuesto"
        sino "probablemente primo"
```

El costo del algoritmo es $O(k \log^3 n)$.

Por el teorema 2.3.2, la probabilidad de que el algoritmo afirme que un número compuesto dado es probablemente primo es menor que $(1/4)^k$.

Cuando $k = 100$, esta probabilidad es menor que 10^{-60}, ¡y sin asumir hipótesis adicionales!

2.4 La Hipótesis de Riemann Extendida

La *Hipótesis de Riemann* es uno de los problemas más importantes de la matemática de los últimos tiempos. La conjetura, establecida por Riemann en 1859, tiene que ver con la ubicación de los ceros de una función compleja, la *función zeta de Riemann*, que relaciona las estructuras aditiva y multiplicativa de los números enteros. El teorema del número primo, que mencionamos anteriormente, es equivalentemente al hecho de que la

función zeta no posee ceros en una recta del plano complejo, y por lo tanto un caso especial. Numerosos esfuerzos se han realizado por demostrar la hipótesis de Riemann; en el área de la computación, se han hallado una gran cantidad de ceros; los resultados hasta ahora obtenidos reafirman la hipótesis. La Hipótesis de Riemann posee algunas generalizaciones, entre ellas la *Hipótesis de Riemann Extendida (HRE)* que establece la ubicación de los ceros de las L-series de Dirichlet (ver [3]). Asumiendo la Hipótesis de Riemann Extendida pueden probarse muchos resultados, entre ellos el siguiente, debido a E.Bach (ver [6], [8]):

Teorema 2.4.1 (HRE) Sea G un subconjunto de $\mathbb{Z}_n^*$ cerrado con respecto a la multiplicación. Si $G \neq \mathbb{Z}_n^*$, entonces existe $a \in \mathbb{Z}_n^* \setminus G$ tal que $a < 2\log^2 n$.

Con esto se demuestra que:

Teorema 2.4.2 (HRE) Si n es compuesto e impar, entonces existe $1 \leq a < 2\log^2 n$ tal que n no es un pseudoprimo en relación a la base a.

Luego, para determinar la primalidad de un número, basta modificar el test de Miller-Rabin (algoritmo 2.3.1) para realizar las pruebas sobre los números $1 < a < k = 2\log^2 n$ (en realidad, el test original de Miller, publicado en 1976, estaba formulado de esta manera, y fue más tarde modificado por Rabin). Luego el costo total del algoritmo es igual a

$$O(\log^5 n),$$

y el algoritmo es polinomial. Además, podemos reemplazar en el algoritmo la salida "probablemente primo" por "primo". Sin embargo, la certeza de la afirmación "primo" al final del algoritmo depende de un resultado aún no demostrado.

2.5 Test de Solovay-Strassen

El test probabilístico de Solovay-Strassen fue el primer test popularizado con el advenimiento de la criptografía de clave pública, en particular del RSA (capítulo 5). Aunque ampliamente superado por el test de Miller-Rabin, lo presentamos por su importancia teórica e histórica.

Definición 2.5.1 Si $(a,n) = 1$ y $a^{(n-1)/2} \equiv \left(\frac{a}{n}\right) \pmod{n}$, decimos que n es un *pseudoprimo de Euler* con respecto a la base a.

Por el criterio de Euler (teorema 1.10.2), todo primo es un pseudoprimo de Euler con respecto a cualquier base a.

Teorema 2.5.1 Si n es compuesto, entonces n es pseudoprimo de Euler a lo más con respecto al 50% de los elementos de $\mathbb{Z}_n^*$.

Prueba. Por un argumento similar a la prueba del teorema 2.2.1, siendo ambos lados de la igualdad

$$a^{(n-1)/2} \equiv \left(\frac{a}{n}\right) (\bmod n)$$

completamente multiplicativos, para probar el resultado buscado basta probar que existe $a \in \mathbb{Z}_n^*$ tal que $a^{(n-1)/2} \not\equiv \left(\frac{a}{n}\right) (\bmod n)$.

Supongamos entonces que se tiene $a^{(n-1)/2} \equiv \left(\frac{a}{n}\right) (\bmod n)$, para todo a con $(a,n) = 1$. Claramente n es un número de Carmichael (basta elevar al cuadrado la igualdad), así que es libre de cuadrados y posee más de dos factores primos.

Escribamos $n = rs$ con $(r,s) = 1$. Afirmo que

$$a^{(n-1)/2} \equiv 1 (\bmod n) \text{ si } (a,n) = 1. \tag{2.2}$$

Por reducción al absurdo, supongamos que existe a tal que $(a,n) = 1$ y

$$a^{(n-1)/2} \equiv -1 (\bmod n).$$

Por el teorema chino del resto, existe b entero tal que

$$b \equiv 1 (\bmod r), \qquad b \equiv a (\bmod s);$$

luego $b^{(n-1)/2} \equiv 1 (\bmod r)$ y $b^{(n-1)/2} \equiv a^{(n-1)/2} \equiv -1 (\bmod s)$, de donde $b^{(n-1)/2} \not\equiv \pm 1 (\bmod n)$, una contradicción.

Sea ahora $n = pr$ con p primo y $(p,r) = 1$. Sea g un residuo no cuadrático módulo p. Nuevamente, el teorema chino del resto nos da a tal que

$$a \equiv g (\bmod p), \qquad a \equiv 1 (\bmod r).$$

Luego

$$\left(\frac{a}{n}\right) = \left(\frac{a}{pr}\right) = \left(\frac{a}{p}\right)\left(\frac{a}{r}\right) = -1\left(\frac{1}{r}\right) \equiv -1 (\bmod n),$$

que junto con (2.2) nos da $\left(\frac{a}{n}\right) = -1 \not\equiv 1 \equiv a^{(n-1)/2} (\bmod n)$.

Algoritmo 2.5.1 (Solovay-Strassen) Este algoritmo determina si un entero dado n es compuesto con certeza, o si es primo con una posibilidad de error de $(1/2)^k$.

Entrada: n entero impar k parámetro de seguridad
$i = 1$
repetir
escogemos aleatoriamente a con $1 < a < n$
si $(a, n) = 1 \wedge a^{n-1} \equiv (\frac{a}{n}) (\bmod n)$
$i = i + 1$
sino
$i = 0$
mientras $(i > 0 \wedge i \leq k)$
Salida: si $i = 0$ "compuesto" sino "probablemente primo"

Por el teorema anterior, la probabilidad de que este algoritmo declare un número compuesto como "probablemente primo" es menor que $(1/2)^k$.

Como sabemos, el costo de calcular el símbolo de Jacobi es $O(\log^3 n)$. Luego el costo total del algoritmo de Solovay-Strassen es $O(k \log^3 n)$.

2.6 Pseudoprimos de Lucas y de Fibonacci

Vamos ahora a estudiar un test de primalidad basado en la sucesión de Fibonacci $(u_n)_{j \geq 0}$, definida por

$$u_0 = 1, \quad u_1 = 1, \quad \text{y} \quad u_j = u_{j-1} + u_{j-2} \text{ para } j \geq 2;$$

y una generalización importante, debida a Lucas (1878).

Teorema 2.6.1 Si n es primo, entonces

$$u_{n-\varepsilon_n} \equiv 0 (\bmod n), \tag{2.3}$$

donde $\varepsilon_n = 1$ cuando $n \equiv \pm 1 (\bmod 5)$, $\varepsilon_n = -1$ cuando $n \equiv \pm 2 (\bmod 5)$, y $\varepsilon_n = 0$ cuando $n \equiv 0 (\bmod 5)$.

Nótese que la sucesión ε_n es el símbolo de Jacobi $\varepsilon_n = \left(\frac{n}{5}\right)$.

Definición 2.6.1 Decimos que un número compuesto es un *pseudoprimo de Fibonacci* si se da (2.3).

Entrada: n entero impar
k parámetro de seguridad

$i = 1$

repetir

  escogemos aleatoriamente
  a con $1 < a < n$

  si $(a, n) = 1 \wedge a^{n-1} \equiv (\frac{a}{n}) (\bmod n)$

    $i = i + 1$

  sino

    $i = 0$

mientras $(i > 0 \wedge i \leq k)$

Salida: si $i = 0$ "compuesto"
sino "probablemente primo"

Por el teorema anterior, la probabilidad de que este algoritmo declare un número compuesto como "probablemente primo" es menor que $(1/2)^k$.

Como sabemos, el costo de calcular el símbolo de Jacobi es $O(\log^3 n)$. Luego el costo total del algoritmo de Solovay-Strassen es $O(k \log^3 n)$.

2.6 Pseudoprimos de Lucas y de Fibonacci

Vamos ahora a estudiar un test de primalidad basado en la sucesión de Fibonacci $(u_n)_{j \geq 0}$, definida por

$$u_0 = 1, \quad u_1 = 1, \quad \text{y} \quad u_j = u_{j-1} + u_{j-2} \text{ para } j \geq 2;$$

y una generalización importante, debida a Lucas (1878).

Teorema 2.6.1 Si n es primo, entonces

$$u_{n-\varepsilon_n} \equiv 0 (\bmod n), \tag{2.3}$$

donde $\varepsilon_n = 1$ cuando $n \equiv \pm 1 (\bmod 5)$, $\varepsilon_n = -1$ cuando $n \equiv \pm 2 (\bmod 5)$, y $\varepsilon_n = 0$ cuando $n \equiv 0 (\bmod 5)$.

Nótese que la sucesión ε_n es el símbolo de Jacobi $\varepsilon_n = \left(\frac{n}{5}\right)$.

Definición 2.6.1 Decimos que un número compuesto es un *pseudoprimo de Fibonacci* si se da (2.3).

Demostración del teorema 2.6.2. Supongamos que p es un primo impar con $\left(\frac{\Delta}{p}\right) = -1$. Entonces Δ no es un cuadrado en $\mathbb{Z}_p$, así que el polinomio $x^2 - ax + b$ es irreducible sobre $\mathbb{Z}_p$ (observe que

$$x^2 - ax + b \equiv (x - 2^{-1}a)^2 - (2^{-1})^2\Delta(\bmod (f(x), n))$$

es reducible en R,

$$x^2 - ax + b \equiv (x - 2^{-1}a + c)(x - 2^{-1}a - c)(\bmod (f(x), n))$$

si $\Delta \equiv c^2(\bmod n)$). Por lo tanto, $R = \mathbb{Z}_p[x]/(x^2 - ax + b)$ es isomorfo al campo finito $\mathbb{F}_{p^2}$ con p^2 elementos. El subcampo $\mathbb{Z}_p$ $(= \mathbb{F}_p)$ está formado por los representantes $i + jx$ con $j = 0$. Antes de proseguir con la prueba, necesitamos un poco más de información de $\mathbb{F}_{p^2}$:

Proposición 2.6.3 En $\mathbb{F}_{p^2}$, la función $\sigma : \mathbb{F}_{p^2} \to \mathbb{F}_{p^2}$ definida por $\sigma(u) = u^p$, llamada *automorfismo de Frobenius*, posee las siguientes propiedades:

(i) $\sigma(u + v) = \sigma(u) + \sigma(v)$, $u, v \in \mathbb{F}_{p^2}$.

(ii) $\sigma(uv) = \sigma(u)\sigma(v)$, $u, v \in \mathbb{F}_{p^2}$.

(iii) $\sigma(u) = u$ si y sólo si $u \in \mathbb{Z}_p$.

Demostración.

(i) Recordemos que $R = \mathbb{F}_{p^2}$ es un dominio de característica p. Luego, por el teorema del binomio de Newton

$$\sigma(u + v) = \sum_{k=0}^{p} \binom{p}{k} u^{p-k}v^k = u^p + v^p = \sigma(u) + \sigma(v)$$

pues para $0 < k < p$, $p \left| \binom{p}{k} \right.$.

(ii) $\sigma(uv) = (uv)^p = u^p v^p = \sigma(u)\sigma(v)$.

(iii) Escribimos $u = i + jx$, $0 \leq i, j \leq p - 1$. Por el pequeño teorema de Fermat

$$i^p \equiv i(\bmod p), \text{ de donde } i^p = i \text{ en } R.$$

De (i) y (ii)

$$\sigma(u) = \sigma(i) + \sigma(j)\sigma(x) = i + jx^p$$

en R. En particular si $j = 0$ $(u \in \mathbb{Z}_p)$ entonces $\sigma(u) = u$.
Consideremos ahora el polinomio $q(x) = x^p - x$ sobre R (no confundir el polinomio

y el elemento $x \in R$). Como el grado de q es p, no pueden existir más de p raíces de q en el campo $\mathbb{F}_{p^2}$. Por otro lado, $\mathbb{Z}_p$ es un conjunto de p elementos, todos raíces de p. Por lo tanto

$$q(u) = 0 \text{ si y sólo si } u \in \mathbb{Z}_p,$$

esto es, $\sigma(u) = u$ si y sólo si $u \in \mathbb{Z}_p$.

Veamos ahora que, efectivamente, σ es un automorfismo. Como $\mathbb{F}_{p^2}$ es finito y $\sigma : \mathbb{F}_{p^2} \to \mathbb{F}_{p^2}$, basta probar que σ es inyectiva. Sea $u \in \mathrm{Ker}(\sigma)$: $\sigma(u) = 0$, i.e.

$$i + jx^p = 0$$

en R. Si $j = 0$, entonces $i = 0$ y $u = 0$. Si $j \neq 0$, escribimos $x^p = -ij^{-1}$ y luego, como $F_{p^2}^*$ tiene orden $p^2 - 1$,

$$x \equiv x^{p^2} \equiv (-ij^{-1})^p \equiv -ij^{-1} (\bmod (f(x), p)),$$

de donde $x \in \mathbb{Z}_p$, absurdo.

Demostración del teorema 2.6.2 (continuación). Creamos el campo $\mathbb{F}_{p^2}$ para proporcionar raíces para $x^2 - ax + b$, de las que carece $\mathbb{Z}_p$. Estas raíces son exactamente x y $a - x$ $(= a + (p-1)x)$. Ahora observe que la imagen de una raíz de $x^2 - ax + b$ mediante σ es una raíz de σ (basta aplicar σ a la ecuación $f(x) \equiv 0$); como x y $a - x$ no pertenecen a $\mathbb{Z}_p$, entonces σ deberá permutar las raíces, i.e.

$$\begin{aligned} x^p &\equiv a - x (\bmod (f(x), p)), \\ (a-x)^p &\equiv x (\bmod (f(x), p)). \end{aligned} \tag{2.7}$$

De donde

$$x^{p+1} - (a-x)^{p+1} \equiv x(a-x) - (a-x)x \equiv 0 (\bmod (f(x), p)),$$

y por lo tanto $U_{p+1} \equiv 0 (\bmod p)$.

Veamos ahora el caso en que $\left(\frac{\Delta}{p}\right) = 1$. En este caso $x^2 - ax + b$ posee dos raíces en $\mathbb{Z}_p$; sean u y v dichas raíces (que son distintas), escribimos

$$x^2 - ax + b = (x-u)(x-v) \equiv 0 (\bmod (f(x), p)) \tag{2.8}$$

(lo que muestra que $R = \mathbb{Z}_p[x]/(x^2 - ax + b)$ no es un dominio). Consideremos nuevamente el polinomio $q(x) = x^p - x$ en R. Como antes, $\mathbb{Z}_p$ es un conjunto de raíces de q, por lo que podemos escribir

$$q(x) = x(x-1)(x-2)\cdots(x-(p-1)).$$

Pero entonces el producto (2.8) está contenido en el producto anterior, y entonces $q(x) = 0$, esto es,

$$x^p = x(\bmod (f(x), p)).$$

Utilizando las propiedades (i) y (ii) de la proposición 2.6.3 que siguen siendo válidas en R (sin necesidad de que sea un campo) tenemos, para todo $u = i + jx$ en R:

$$\sigma(u) = u^p = i + jx^p \equiv i + jx(\bmod (f(x), p)).$$

En particular nos interesan las identidades

$$\begin{aligned} x^p &\equiv x(\bmod (f(x), p)), \\ (a - x)^p &\equiv a - x(\bmod (f(x), p)). \end{aligned} \tag{2.9}$$

Note además que la hipótesis de que $(p, b) = 1$ implica que x y $a - x$ son invertibles en R, pues

$$x(a - x) \equiv b(\bmod (f(x), p)).$$

Por lo tanto $x^p = (a - x)^p = 1$ en R, y $U_{p-1} \equiv 0(\bmod p)$. Esto concluye la demostración.

2.7 Test de Grantham-Frobenius

Analizando la prueba del teorema 2.6.2, notamos que las ecuaciones (2.7) y (2.9) son de vital importancia. Pensamos entonces en extraer dicha información directamente para crear un test más efectivo:

Definición 2.7.1 Sean a, b enteros con $\Delta = a^2 - 4b$ no cuadrado. Decimos que un entero compuesto n con $(n, 2b\Delta) = 1$ es un *pseudoprimo de Frobenius con respecto a* $f(x) = x^2 - ax + b$ si

$$x^n \equiv \begin{cases} a - x(\bmod (f(x), n)), & \text{si } \left(\frac{\Delta}{n}\right) = -1 \\ x(\bmod (f(x), n)), & \text{si } \left(\frac{\Delta}{n}\right) = 1. \end{cases}$$

Veamos el caso $\left(\frac{\Delta}{n}\right) = -1$: cambiando x por $a - x$ en la congruencia indicada arriba, tenemos

$$(a - x)^n \equiv x(\bmod (f(a - x), n),$$

pero $f(a - x) = (a - x)^2 - a(a - x) + b = f(x)$ de donde

$$(a - x)^n \equiv x(\bmod (f(x), n).$$

Realizando un cálculo análogo en el caso $\left(\frac{\Delta}{n}\right) = 1$, obtenemos

$$(a-x)^n \equiv \begin{cases} x(\bmod (f(x),n)), & \text{si } \left(\frac{\Delta}{n}\right) = -1, \\ a - x(\bmod (f(x),n)), & \text{si } \left(\frac{\Delta}{n}\right) = 1 \end{cases}$$

y por lo tanto un pseudoprimo de Frobenius respecto a $f(x)$ es también un pseudoprimo de Lucas respecto a $f(x)$.

Teorema 2.7.1 Sean a, b enteros con $\Delta = a^2 - 4b$ no cuadrado y n un entero compuesto con $(n, 2b\Delta) = 1$. Entonces n es un pseudoprimo de Frobenius con respecto a $x^2 - ax + b$ si y sólo si

$$U_{n-(\frac{\Delta}{n})} \equiv 0(\bmod n) \quad \text{y} \quad V_{n-(\frac{\Delta}{n})} \equiv \begin{cases} 2b(\bmod (f(x),n)), & \text{si } \left(\frac{\Delta}{n}\right) = -1 \\ 2(\bmod (f(x),n)), & \text{si } \left(\frac{\Delta}{n}\right) = 1. \end{cases} \tag{2.10}$$

Prueba. Sea $f(x) = x^2 - ax + b$. Consideremos la identidad

$$2x^m \equiv (2x-a)U_m + V_m(\bmod (f(x),n)). \tag{2.11}$$

Supongamos que se cumple (2.10). En caso que $\left(\frac{\Delta}{n}\right) = -1$, de (2.11),

$$2x^{n+1} \equiv (2x-a)U_{n+1} + V_{n+1} \equiv (2x-a)0 + 2b \equiv 2b(\bmod (f(x),n)),$$

de donde $x^{n+1} \equiv b(\bmod (f(x),n))$ y como $x(a-x) \equiv b$,

$$bx^n \equiv x^{n+1}(a-x) \equiv b(a-x)(\bmod (f(x),n))$$

y por lo tanto $x^n \equiv (a-x)(\bmod (f(x),n))$. Si en cambio, $\left(\frac{\Delta}{n}\right) = 1$, de (2.11)

$$2x^{n-1} \equiv (2x-a)U_{n-1} + V_{n-1} \equiv (2x-a)0 + 2 \equiv 2(\bmod (f(x),n)),$$

entonces $x^{n-1} \equiv 1(\bmod (f(x),n))$ y luego $x^n \equiv x(\bmod (f(x),n))$. Por lo tanto n es un pseudoprimo de Frobenius respecto a $x^2 - ax + b$.

Supongamos ahora que n es un pseudoprimo de Frobenius respecto a $f(x)$. Entonces, sabemos que es un pseudoprimo de Lucas con respecto a $f(x)$, es decir

$$U_{n-(\frac{\Delta}{n})} \equiv 0(\bmod n).$$

De la identidad (2.11),

$$2x^{n-(\frac{\Delta}{n})} \equiv V_{n-(\frac{\Delta}{n})}(\bmod(f(x),n)).$$

Si $\left(\frac{\Delta}{n}\right) = -1$, entonces $x^{n+1} \equiv (a-x)x \equiv b(\bmod(f(x),n))$ y por lo tanto

$$V_{n+1} \equiv 2b(\bmod n).$$

Si $\left(\frac{\Delta}{n}\right) = 1$, como x es invertible módulo $(f(x),n)$, entonces $x^{n-1} \equiv 1(\bmod(f(x),n))$ y

$$V_{n-1} \equiv 2(\bmod n).$$

Ejemplo 2.7.1 El primer pseudoprimo con respecto a $x^2 - x - 1$ es $n = 5777$.

2.8 Implementación de los tests de Lucas y Frobenius

Para implementar los tests de Lucas y Frobenius, debemos buscar una forma eficiente de calcular

$$U_{n-(\frac{\Delta}{n})}(\bmod n) \qquad \text{y} \qquad V_{n-(\frac{\Delta}{n})}(\bmod n),$$

ya que evaluar las sucesiones por definición no es una buena opción (evaluar término a término a través de las recurrencias (2.5) toma alrededor de n iteraciones, tiempo exponencial). Por ello introduciremos nuevas fórmulas de recurrencia para calcular las sucesiones requeridas, más apropiadas para la implementación en computadora. Deduciremos estas relaciones por etapas, partiendo del caso más simple.

Sea $\Delta = a^2 - 4b$ no cuadrado, y consideremos las sucesiones (U_j), (V_j).

Veamos que basta trabajar con una de las dos sucesiones, por ejemplo con (V_j) puesto que podemos recuperar (U_j) a partir de

$$U_m = \Delta^{-1}(2V_{m+1} - aV_m).$$

Además, si $0 \leq j \leq k$, entonces

$$V_{j+k} = V_jV_k - b^jV_{k-j}.$$

Supongamos que $b = 1$. Si consideramos el cálculo de los elementos $k = j$, $k = j+1$ tenemos

$$V_{2j} = V_j^2 - 2, \quad V_{2j+1} = V_jV_{j+1} - a.$$

Luego, empezando de V_0, V_1 podemos llegar a cualquier par V_m, V_{m+1}. Veamos el caso $m = 74$:

$$0,1 \to 1,2 \to 2,3 \to 4,5 \to 9,10 \to 18,19 \to 37,38 \to 74,75.$$

En esta secuencia de pares observamos dos tipos de movimientos:

$$(a, a+1) \to (2a, 2a+1),$$
$$(a, a+1) \to (2a+1, 2a+2).$$

¿Cómo hallar la secuencia? Podemos utilizar el desarrollo binario de m. En este caso

$$74 = (1001010)_2.$$

Un cero significa un movimiento del primer tipo y un uno, del segundo. Con todas estas observaciones, estamos listos para construir un algoritmo para calcular $V_m (\bmod n)$ cuando $b = 1$:

Algoritmo 2.8.1 (Cadena de Lucas) Este algoritmo calcula el par (x_m, x_{m+1}) de una sucesión para la cual se conoce una regla para calcular x_{2j} a partir de x_j (denotada $x_{2j} = x_j * x_j$) y otra regla para calcular x_{2j+1} a partir de x_j, x_{j+1} (denotada $x_{2j+1} = x_j \circ x_{j+1}$), además de los valores iniciales x_0, x_1.

Entrada:	$n = (n_0 n_1 \ldots n_{B-1})_2$ entero representado en base 2 x_0, x_1 : valores iniciales de la sucesión reglas $x_{2j} = x_j * x_j$, $x_{2j+1} = x_j \circ x_{j+1}$ de cálculo
$(u, v) = (x_0, x_1)$	
para $((B > j \geq 0))$ [en orden descendente]	
	si $(n_j = 1)$
	$(u, v) = (u \circ v, v * v)$
	sino
	$(u, v) = (u * u, u \circ v)$
Salida:	$(u, v) = (x_n, x_{n+1})$

Observe que, en cada paso del algoritmo, $u = x_j$, $v = x_{j+1}$ para algún entero j.

Vamos a relacionar las sucesiones (V_j) generadas para distintos valores de a, b; para resaltar esto denotamos $V_m = V_m(a, b)$. Reduciremos el cálculo de una sucesión cualquiera a una sucesión $V_j(a, b)$ con $b = 1$.

Si $a = cd$, $b = d^2$ entonces es fácil ver por inducción que

$$V_m(cd, d^2) = d^m V_m(c, 1).$$

Si suponemos apenas que b es un cuadrado, $b = d^2$, y que $(n, b) = 1$, entonces de lo anterior

$$V_m(a, d^2) \equiv d^m V_m(ad^{-1}, 1) (\bmod n),$$

donde d^{-1} es la inversa de d módulo n. Ahora, para el caso más general, donde b no es un cuadrado, observe que

$$V_{2m}(a,b) = V_m(a^2 - 2b, b^2),$$

(donde "Δ" al lado derecho es un cuadrado) y que si

$$A \equiv b^{-1}V_2(a,b) \equiv a^2b^{-1} - 2 (\mathrm{mod}\, n),$$

entonces

$$V_{2m}(a,b) \equiv b^m V_m(A,1) (\mathrm{mod}\, n).$$

Análogamente

$$U_{2m}(a,b) \equiv ab^{m-1}U_m(A,1) (\mathrm{mod}\, n).$$

Luego

$$U_{2m}(a,b) \equiv (a\Delta)^{-1}b^{m+1}(2V_{m+1}(A,1) - AV_m(A,1)) (\mathrm{mod}\, n).$$

Para realizar los tests de Lucas y Frobenius, necesitamos calcular U_{2m} para $2m = n - \left(\frac{\Delta}{n}\right)$. Todo lo anterior se resume en el siguiente

Teorema 2.8.1 Supongamos que a, b, Δ son tales que

$$A \equiv a^2b^{-1} - 2 (\mathrm{mod}\, n),$$

y n compuesto, $(n, 2ab\Delta) = 1$. Entonces n es un pseudoprimo de Lucas con respecto a $x^2 - ax + b$ si y sólo si

$$AV_{\frac{1}{2}(n-(\frac{\Delta}{n}))}(A,1) \equiv 2V_{\frac{1}{2}(n-(\frac{\Delta}{n}))+1}(A,1) (\mathrm{mod}\, n);$$

y n es un pseudoprimo de Frobenius con respecto a $x^2 - ax + b$ si y sólo si se cumple lo anterior y además

$$b^{(n-1)/2}V_{\frac{1}{2}(n-(\frac{\Delta}{n}))}(A,1) \equiv 2 (\mathrm{mod}\, n).$$

Esta formulación de los criterios de Lucas y Frobenius nos permite utilizar el algoritmo de la cadena de Lucas para comprobarlos.

Algoritmo 2.8.2 (Tests de Lucas y Frobenius) Este algoritmo comprueba si un entero dado n es pseudoprimo de Lucas y Frobenius con respecto a $f(x) = x^2 - ax + b$.

Entrada: n, a, b enteros, $n > 1$, $\Delta = a^2 - 4b$ no cuadrado $(n, 2ab\Delta) = 1$
$A = (a^2 b^{-1} - 2) \bmod n$ $m = \left(n - \left(\frac{\Delta}{n}\right)\right)/2$
usar el algoritmo de la cadena de Lucas para calcular (V_m, V_{m+1}) con valores iniciales $(V_0, V_1) = (2, A)$ y reglas $V_{2j} = V_j^2 - 2 \bmod n, \quad V_{2j+1} = V_j V_{j+1} - A \bmod n$
$pslucas =$ Verdadero $psfrob =$ Verdadero
[Test de Lucas] $pslucas = (AV_m \equiv 2V_{m+1} (\bmod n))$
[Test de Frobenius] $B = b^{(n-1)/2} \bmod n$ $psfrob = pslucas \wedge (BV_m \equiv 2 (\bmod n))$
Salida: ¿n es pseudoprimo de Lucas? $= pslucas$ ¿n es pseudoprimo de Frobenius? $= psfrob$

Capítulo 3

Verificando primalidad de números de forma especial

En este capítulo presentamos una serie de algoritmos para determinar, efectivamente, la primalidad de números de forma especial, o bien números de los cuales se ha obtenido cierta información (veremos cómo). Los candidatos ideales son números presumiblemente primos, números que han superado tests probabilísticos como los presentados en el capítulo 2. Hay que resaltar, además, que no todos los números enteros pueden ser probados utilizando estos algoritmos. Un algoritmo de propósito general se estudia en el siguiente capítulo.

3.1 El test de Pocklington-Lehmer

Necesitamos alguna clase de recíproco del teorema de Fermat.

Teorema 3.1.1 (Pocklington) Supongamos que $n-1 = F \cdot R$, donde F está completamente factorizado, y a es tal que

$$a^{n-1} \equiv 1 (\bmod n) \text{ y } (a^{(n-1)/q} - 1, n) = 1 \text{ para cada primo } q|F. \tag{3.1}$$

Entonces todo divisor d de n cumple

$$d \equiv 1 (\bmod F).$$

Si además $F \geq \sqrt{n}$, entonces n es primo.

Prueba. Basta demostrar el teorema cuando d es primo. Sea q un primo que divide a F. De la condición (3.1)

$$a^{n-1} \equiv 1 (\bmod d) \text{ y } (a^{(n-1)/q} - 1, d) = 1,$$

es decir $a^{(n-1)/q} \not\equiv 1 (\mathrm{mod}\, d)$. Luego

$$\exp_d(a) | n-1 \text{ pero } \exp_d(a) \not| \frac{n-1}{q},$$

de donde si $q^\alpha | n-1$, entonces $q^\alpha | \exp_d(a)$. Por lo tanto $F | \exp_d(a)$ y como $\exp_d(a) | d-1$ siempre, entonces $F | d-1$, como queríamos. Si además $\sqrt{n} \leq F$ entonces de $F | d-1$, tenemos que $F \leq d-1$ y

$$\sqrt{n} \leq F < d,$$

y todo divisor primo de n es $> \sqrt{n}$. Por lo tanto n es primo.

3.1.1 Números de Fermat

Usaremos el teorema de Pocklington para probar la primalidad de números de forma especial.

Proposición 3.1.2 (Proth) Sean $k, l \in \mathbb{Z}$, $l \geq 2$, $3 \not| k$ y $k \leq 2^l - 1$. Entonces $n = k \cdot 2^l + 1$ es primo si y sólo si

$$3^{k \cdot 2^{l-1}} \equiv -1 (\mathrm{mod}\, n).$$

Demostración. Supongamos que $3^{k \cdot 2^{l-1}} \equiv -1 (\mathrm{mod}\, n)$. Hagamos $F = 2^l$, $a = 3$, luego de la hipótesis

$$a^{n-1} = 3^{k \cdot 2^l} \equiv 1 (\mathrm{mod}\, n)$$

y $a^{(n-1)/2} \equiv -1 (\mathrm{mod}\, n)$, de donde

$$(a^{(n-1)/2} - 1, n) = (-3, n) = 1.$$

Luego por el teorema de Pocklington, n es primo.

Recíprocamente, supongamos que n es primo. Veamos que 3 es un resto no cuadrático módulo n. Como $3 \not| k$ y n es primo,

$$n = k \cdot 2^l + 1 \equiv 2 (\mathrm{mod}\, 3).$$

Luego por la ley de reciprocidad cuadrática y el criterio de Euler

$$\left(\frac{3}{n}\right) = \left(\frac{3}{k \cdot 2^l + 1}\right) = \left(\frac{k \cdot 2^l + 1}{3}\right) = \left(\frac{2}{3}\right) = -1,$$

y

$$3^{k \cdot 2^{l-1}} = 3^{(n-1)/2} \equiv \left(\frac{3}{n}\right) \equiv -1 (\mathrm{mod}\, n).$$

Ahora restringimos nuestro estudio al caso $k = 1$

Teorema 3.1.3 Si $p = 2^m + 1$ es primo, entonces $m = 2^n$ con $n \geq 0$.

Prueba. Supongamos que $m = ab$ con $a = \max\{d : d \text{ impar }, d|m\}$. Como $2^b + 1|p$, entonces $p = 2^b + 1$ (p es primo) y por lo tanto $m = b$ y $a = 1$. Luego b es una potencia de 2, como queríamos.

Por el último teorema, basta estudiar los números de la forma $2^{2^n} + 1$:

Definición 3.1.1 Un número de la forma $F_n = 2^{2^n} + 1$ con $n \geq 0$ entero es llamado *número de Fermat*. Los primos de esta forma son llamados *primos de Fermat*.

Antes de intentar verificar si un número de Fermat es primo, podemos hacer una búsqueda de posibles factores. Veamos que estos números de forma especial poseen factores especiales:

Teorema 3.1.4 (Euler) Para $n \geq 2$, todo factor primo p de $F_n = 2^{2^n} + 1$ cumple

$$p \equiv 1 (\bmod 2^{n+2}).$$

Prueba. Sea r un factor primo de F_n y

$$h = \min\{k \in \mathbb{N} : 2^k \equiv 1 (\bmod r)\} = \exp_r(2).$$

Como $2^{2^n} \equiv -1 (\bmod p)$, entonces $h = 2^{n+1}|p - 1$. Como $n \geq 2$, entonces $p \equiv 1 (\bmod 8)$. Por lo tanto 2 es un resto cuadrático módulo p, esto es existe a entero con

$$a^2 \equiv 2 (\bmod p)$$

y esto implica que $(a, p) = 1$ y de lo anterior, $a^{2^{n+1}} \equiv -1 (\bmod p)$ y esto implica a su vez que $\exp_r(a) = 2^{n+2}|p - 1$. Por lo tanto $p \equiv 1 (\bmod 2^{n+2})$.

Una vez hecha la búsqueda de posibles factores para un número de Fermat, sólo nos queda finalmente aplicar el siguiente test:

Proposición 3.1.5 (Pepin) Para todo $m \in \mathbb{Z}$, $m \geq 1$, el número de Fermat $F_m = 2^{2^m} + 1$ es primo si y sólo si

$$3^{(F_m - 1)/2} \equiv -1 (\bmod F_m).$$

Prueba. Basta tomar $k = 1$ y $l = 2^m$ en la proposición 3.1.2.

3.2 El test $n-1$

Ahora tratamos de debilitar la condición $F \geq n^{1/2}$ en el teorema de Pocklington.

Teorema 3.2.1 (Brillhart, Lehmer, Selfridge) Supongamos que $n-1 = FR$, donde F está completamente factorizado, y a cumple (3.1). Supongamos además que $n^{1/3} \leq F < n^{1/2}$. Consideremos la representación de n en la base F, $n = c_2F^2 + c_1F + 1$, donde c_1, c_2 son enteros en $[0, F-1]$. Entonces n es primo si y sólo si $c_1^2 - 4c_2$ no es un cuadrado.

Prueba. Como $n \equiv 1 (\bmod F)$, n se escribe en la forma $c_2F^2 + c_1F + 1$, como se afirmó. Supongamos que n es compuesto. Por el teorema de Pocklington, si p es un factor primo de n,

$$p \equiv 1 (\bmod F)$$

y por lo tanto

$$p > p - 1 \geq F \geq n^{1/3}.$$

Luego, n posee exactamente dos factores primos, p y q:

$$n = pq, \quad p = aF + 1, \quad q = bF + 1, \quad a \leq b.$$

Entonces

$$c_2F^2 + c_1^F + 1 = n = (aF + 1)(bF + 1) = abF^2 + (a + b)F + 1.$$

Probaremos que $c_2 = ab$ y $c_1 = a + b$, de donde concluiremos que

$$c_1^2 - 4c_2 = (a - b)^2,$$

un cuadrado.

Notemos que $F^3 \geq n > abF^2$, de manera que $ab \leq F - 1$. Se sigue que $a + b \leq F - 1$ o bien $a = 1$, $b = F - 1$. En el último caso, $n = (F + 1)((F - 1)F + 1) = F^3 + 1$, contradiciendo el hecho que $F \geq n^{1/3}$. Por lo tanto ab y $a + b$ son enteros positivos y menores que F. Por la unicidad de la representación de n en la base F, $c_2 = ab$ y $c_1 = a + b$, como queríamos.

Recíprocamente, supongamos ahora que $c_1^2 - 4c_2 = u^2$ (un cuadrado). Entonces

$$n = \left(\frac{c_1 + u}{2}F + 1\right)\left(\frac{c_1 - u}{2}F + 1\right).$$

Observe que los dos factores en la última igualdad son enteros, pues $c_1 \equiv u (\bmod 2)$. Además los factores son no triviales, pues $c_2 > 0$ implica que $|u| < c_1$. Por lo tanto, n es compuesto.

Podemos considerar F incluso más pequeño. Ese es el resultado del próximo teorema.

Teorema 3.2.2 (Konyagin-Pomerance) Supongamos que $n \geq 214$, $n-1 = FR$ donde F está completamente factorizado, y existe a que satisface (3.1). Supongamos además que $n^{3/10} \leq F < n^{1/3}$. Sea $n = c_3F^3 + c_2F^2 + c_1F + 1$ la representación de n en la base F, y $c_4 = c_3F + c_2$. Entonces n es primo si y sólo si se cumplen las siguientes condiciones:

(i) $(c_1 + tF)^2 + 4t - 4c_4$ no es un cuadrado para $t = 0, 1, 2, 3, 4, 5$.

(ii) Con u/v el convergente de la fracción continua de c_1/F tal que v es maximal sujeto a $v < F^2/\sqrt{n}$ y con $d = \lfloor c_4v/F + 1/2 \rfloor$, el polinomio $vx^3 + (uF - c_1v)x^2 + (c_4v - dF + u)x - d \in \mathbb{Z}[x]$ no posee raíz integral a tal que $aF + 1$ es factor no trivial de n.

Prueba. Por el teorema de Pocklington, todo factor primo de n es congruente a $1 (\bmod F)$; luego n es compuesto si y sólo si existen enteros positivos $a_1 \leq a_2$ tales que

$$n = (a_1F + 1)(a_2F + 1).$$

Supongamos que n es compuesto y se cumplen (1) y (2). Tenemos

$$n = c_4F^2 + c_1F + 1 = a_1a_2F^2 + (a_1 + a_2)F + 1,$$

y existe $t \geq 0$, entero con

$$a_1a_2 = c_4 - t, \quad a_1 + a_2 = c_1 + tF. \tag{3.2}$$

Como

$$(c_1 + tF)^2 - 4t - 4c_4 = (a_1 - a_2)^2,$$

por (1), $t \geq 6$. De ahí

$$a_2 \geq \frac{a_1 + a_2}{2} \geq \frac{c_1 + 6F}{2} \geq 3F$$

y

$$a_1 < \frac{n}{a_2F^2} \leq \frac{n}{3F^3}. \tag{3.3}$$

Por (3.2) tenemos que

$$t \leq \frac{a_1 + a_2}{F} \leq \frac{a_1a_2 + 1}{F} < \frac{c_4}{F} < \frac{n}{F^3} \tag{3.4}$$

y además

$$a_1c_1 + a_1tF = a_1^2 + c_4 - t. \tag{3.5}$$

Ahora, de la anterior expresión

$$\begin{aligned} a_1u + a_1tv - \frac{c_4v}{F} &= a_1v\left(\frac{u}{v} - \frac{c_1}{F}\right) + (a_1c_1 + a_1tF)\frac{v}{F} - \frac{c_4v}{F} \\ &= a_1v\left(\frac{u}{v} - \frac{c_1}{F}\right) + (a_1^2 + c_4 - t)\frac{v}{F} - \frac{c_4v}{F} \\ &= a_1v\left(\frac{u}{v} - \frac{c_1}{F}\right) + (a_1^2 - t)\frac{v}{F}. \end{aligned} \tag{3.6}$$

Note que (3.3), (3.4) y $t \geq 6$ implican que

$$|a_1^2 - t| < \max\{a_1^2, t\} \leq \max\left\{\frac{1}{9}\left(\frac{n}{F^3}\right)^2, \frac{n}{F^3}\right\} \leq \frac{1}{6}\left(\frac{n}{F^3}\right)^2.$$

Supongamos que $u/v = c_1/F$. Entonces las dos últimas expresiones implican que

$$\left|a_1 u + a_1 t v - \frac{c_4 v}{F}\right| = |a_1^2 - t|\frac{v}{F} < \frac{1}{6}\left(\frac{n}{F^3}\right)^2 \frac{v}{F} < \frac{n^2}{6F^2} \cdot \frac{F^2}{\sqrt{n}} = \frac{n^{3/2}}{6F^5} \leq \frac{1}{6}. \tag{3.7}$$

Si $u/v \neq c_1/F$, sea u'/v' el próximo convergente de la fracción continua de c_1/F, el siguiente a u/v, de modo que

$$v < \frac{F^2}{\sqrt{n}} \leq v', \quad \left|\frac{u}{v} - \frac{c_1}{F}\right| \leq \frac{1}{vv'} \leq \frac{\sqrt{n}}{vF^2}.$$

Luego, de (3.3), (3.6), y (3.7)

$$\left|a_1 u + a_1 t v - \frac{c_4 v}{F}\right| \leq a_1 v \frac{\sqrt{n}}{vF^2} + \frac{1}{6} < \frac{n^{3/2}}{3F^5} + \frac{1}{6} \leq \frac{1}{2}.$$

Sea $d = a_1 u + a_1 t v$, de modo que $|d - c_4 v/F| < 1/2$, lo que implica que $d = \lfloor c_4 v/F + 1/2 \rfloor$. Multiplicando (3.5) por $a_1 v$, tenemos

$$v a_1^3 - c_1 v a_1^2 - a_1^2 t v F - a_1 t v + c_4 a_1 v = 0,$$

y usando $-a_1 t v = u a_1 - d$, tenemos

$$v a_1^3 + (uF - c_1 v) a_1^2 + (c_4 v - dF + u) a_1 - d = 0.$$

Por lo tanto, no se cumple (2), lo que prueba que si n es compuesto, entonces no se cumple (1) o (2).

Supongamos ahora que n es primo. Si $t \in \{0, 1, 2, 3, 4, 5\}$ y $(c_1 + tF)^2 - 4c_4 + 4t = u^2$, con u integral, entonces

$$n = (c_4 - t)F^2 + (c_1 + tF)F + 1 = \left(\frac{c_1 + tF + u}{2}F + 1\right)\left(\frac{c_1 + tF - u}{2}F + 1\right).$$

Como n es primo, esta factorización debe ser trivial, esto es,

$$c_1 + tF - |u| = 0,$$

lo que implica que $c_4 = t$. Pero $c_4 \geq F \geq n^{3/10} \geq 214^{3/10} > 5 \geq t$, absurdo. Por lo tanto, si (1) falla, entonces n será compuesto. Es obvio que si n es primo, se cumple (2).

Reuniendo los resultados obtenidos en los teoremas 3.1.1, 3.2.1 y 3.2.2, construimos el siguiente algoritmo.

Algoritmo 3.2.1 (El test $n-1$) Supongamos que tenemos un entero $n \geq 214$ tal que $n-1 = FR$ con F completamente factorizado, y $F \geq n^{3/10}$. Este algoritmo probabilístico trata de decidir si n es primo o no. Sin embargo, a pesar de ser probabilístico, la salida del algoritmo es incondicionalmente cierta.

Entrada: $n \geq 214$ de la forma $n-1 = FR$, $F \geq n^{3/10}$
factores primos de F

continuar = Verdadero
primo = Verdadero

mientras *continuar*
- escoger aleatoriamente $2 \leq a \leq n-2$
- si $a^{n-1} \not\equiv 1 (\bmod n)$
  - *primo* = Falso
  - *continuar* = Falso
- sino
  - para (q primo, $q|F$)[probando al candidato]
    - $g = ((a^{(n-1)/q} \bmod n) - 1, n)$
    - si $1 < g < n$ [g sería factor de n]
      - *primo* = Falso
      - *continuar* = Falso
    - si $(1 < g \leq n)$ [si $g = n$ el candidato es inservible]
      - terminar bucle "para"
  - si $(g = 1)$ [se cumple (3.1)]
    - *continuar* = Falso

si *primo*
- [si $F \geq n^{1/2}$ a estas alturas del algoritmo, n es primo]
- si $n^{1/3} \leq F < n^{1/2}$
  - Escribir $n = c_2F^2 + c_1F + 1$ (base F)
  - si $c_1^2 - 4c_2$ es un cuadrado
    - *primo* = Falso
- si $n^{3/10} \leq F < n^{1/3}$
  - si no se cumplen las condiciones del teorema 3.2.2
    - *primo* = Falso

Salida: Respuesta de ¿n es primo? en *primo*

3.3 El test $n+1$

Es evidente la utilidad que tiene el test $n-1$ para números especiales como lo son los números de Fermat, para los cuales la factorización de $n-1$ es trivial. Sin embargo, hay otros números para los cuales resulta más fácil factorizar $n+1$ y no $n-1$: un ejemplo tal lo dan los números de Mersenne, que son de la forma $n = 2^m - 1$.

Para construir el test $n+1$ reintroducimos las sucesiones de Lucas estudiadas en el capítulo anterior.

3.3.1 El test de Lucas

Sean $a, b \in \mathbb{Z}$, y

$$f(x) = x^2 - ax + b, \quad \Delta = a^2 - 4b. \tag{3.8}$$

Recordemos las sucesiones

$$U_k = \frac{x^k - (a-x)^k}{x-(a-x)} (\bmod f(x)), \qquad V_k = x^k + (a-x)^k (\bmod f(x)),$$

las cuales son sucesiones de números enteros.

Definición 3.3.1 Sea $n \in \mathbb{N}$, $(n, 2b\Delta) = 1$. El *rango de apariencia* de n, denotado $r_f(n)$, es el menor entero positivo r tal que

$$U_r \equiv 0 (\bmod n).$$

Proposición 3.3.1 Si $(n, 2b\Delta) = 1$, entonces $U_j \equiv 0 (\bmod n)$ si y sólo si $j \equiv 0 (\bmod r_f(n))$.

Prueba. Hagamos $y = a - x$. Observe que si $k|j$, trivialmente $U_k|U_j$ pues $j = ki$ implica

$$\begin{aligned} \frac{x^j - y^j}{x-y} &= \frac{x^{ki} - y^{ki}}{x-y} = \frac{x^k - y^k}{x-y}[(x^k)^{i-1} + (x^k)^{i-2}y^k + \cdots + x^k(y^k)^{i-2} + (y^k)^{i-1}] \\ &= \frac{x^k - y^k}{x-y} q(x). \end{aligned}$$

de donde

$$U_j \equiv U_k \, q(x) (\bmod f(x)),$$

con U_j, U_k enteros. Luego $q(x) (\bmod f(x))$ es un entero y por lo tanto $U_k|U_j$.
De lo anterior se sigue que si $j \equiv 0 (\bmod r_f(n))$, esto es si $r_f(n)|j$, entonces $U_{r_f(n)}|U_j$, pero además $n|U_{r_f(n)}$ y por lo tanto

$$U_j \equiv 0 (\bmod n).$$

Ahora observemos que si $U_k \equiv 0(\bmod n)$, $U_j \equiv 0(\bmod n)$ con $k \geq j$ entonces $U_{k-j} \equiv 0(\bmod n)$. Trabajamos en el anillo

$$R = \mathbb{Z}_n[x]/(x^2 - ax + b).$$

Observe que $xy = b$ en R, así que x e y son invertibles en R. Ahora

$$\frac{x^k - y^k}{x - y} = x^j \left(\frac{x^{k-j} - y^{k-j}}{x - y}\right) + y^{k-j} \left(\frac{x^j - y^j}{x - y}\right);$$

y reduciendo módulo $(f(x), n)$

$$U_k \equiv x^j U_{k-j} + y^{k-j} U_j (\bmod (f(x), n)),$$

de donde como $U_k \equiv 0 \equiv U_j(\bmod (f(x), n))$ e y es invertible en R, entonces

$$U_{k-j} \equiv 0(\bmod (f(x), n)).$$

Sea ahora j arbitrario tal que $U_j \equiv 0(\bmod n)$. Escribimos

$$j = q \cdot r_f(n) + s, \text{ donde } 0 \leq s < r_f(n).$$

Entonces $s = j - q \cdot r_f(n)$ cumple $U_s \equiv 0(\bmod n)$ por todo lo anterior y por lo tanto $s = 0$ (por la minimalidad de $r_f(n)$), esto es, $r_f(n) | j$.

Una consecuencia inmediata de lo anterior y el teorema 2.6.2 es la siguiente:

Teorema 3.3.2 Sean f, Δ como en (3.8) y p un primo tal que $p \nmid 2b\Delta$. Entonces

$$r_f(p) \left| p - \left(\frac{\Delta}{p}\right)\right..$$

Teorema 3.3.3 (Morrison) Sean f, Δ como en (3.8) y $n \in \mathbb{N}$ tal que $(n, 2b) = 1$, $\left(\frac{\Delta}{n}\right) = -1$. Si F es un divisor de $n + 1$ y

$$U_{n+1} \equiv 0(\bmod n), \quad (U_{(n+1)/q}, n) = 1, \text{ para cada primo } q | F. \tag{3.9}$$

Entonces para todo primo $p | n$, tenemos que $p \equiv \left(\frac{\Delta}{p}\right)(\bmod F)$.
Si además $F > \sqrt{n} + 1$, entonces n es primo.

Prueba. Sea p un factor primo de n. La condición (3.9) implica que si q es un primo que divide a F,

$$U_{n+1} \equiv 0(\bmod p), \quad (U_{(n+1)/q}, p) = 1,$$

es decir $U_{(n+1)/q} \not\equiv 0(\bmod p)$, de donde

$$r_f(p)|n+1 \text{ pero } r_f(p) \not| \frac{n+1}{q}.$$

De aquí, si q es primo y $q^\alpha|F$, entonces $q^\alpha|n+1$ y esto implica que $q^\alpha|r_f(p)$. Por lo tanto $F|r_f(p)$. Por el teorema 3.3.2, $p \equiv \left(\frac{\Delta}{p}\right) (\bmod F)$. Si además $F > \sqrt{n}+1$, entonces $F|p - \left(\frac{\Delta}{p}\right)$ implica

$$F \leq p - \left(\frac{\Delta}{p}\right) \leq p+1$$

de donde

$$\sqrt{n} < F - 1 \leq p,$$

y por lo tanto n es primo.

Así como trabajamos con la sucesión (U_j), podemos dar un test de primalidad utilizando para ello la sucesióm (V_j):

Teorema 3.3.4 Sean f, Δ como en (3.8), $n \in \mathbb{N}$, $(n, 2b) = 1$ y $\left(\frac{\Delta}{n}\right) = -1$. Si F es par, $F|n+1$ y

$$V_{F/2} \equiv 0(\bmod n), \quad (V_{F/2q}, n) = 1, \text{ para todo primo impar } q|F, \tag{3.10}$$

entonces todo divisor primo p de n cumple $p \equiv \left(\frac{\Delta}{p}\right) (\bmod F)$. Si además $F > \sqrt{n}+1$, entonces n es primo.

Prueba. Supongamos que p es un primo impar, $p|U_m$, $p|V_m$, entonces de las definiciones de U_m y V_m

$$x^m \equiv (a-x)^m (\bmod (f(x), p)), \qquad x^m \equiv -(a-x)^m (\bmod (f(x), p)),$$

de donde $x^m \equiv 0(\bmod (f(x), p))$, y

$$b^m \equiv (x(a-x))^m \equiv 0(\bmod (f(x), p)),$$

es decir, $p|b$. Como $(n, b) = 1$ concluimos que si $p|n$, entonces p no pude dividir a U_m y V_m simultáneamente. Luego $U_{2m} = U_m V_m$ implica

$$(U_{2m}, n) = (U_m, n) \cdot (V_m, n).$$

Por la primera condición de (3.10), tenemos que

$$U_F = U_{F/2}V_{F/2} \equiv 0(\bmod n)$$

y

$$n = (U_F, n) = (U_{F/2}, n) \cdot (V_{F/2}, n) = (U_{F/2}, n) \cdot n,$$

de donde $(U_{F/2}, n) = 1$.
Ahora supongamos que q es un primo impar, $q|F$. Tenemos que $U_{F/q} = U_{F/2q} \cdot V_{F/2q}$ es coprimo con n, pues $U_{F/2q}|U_{F/2}$ implica que $(U_{F/2q}, n) = 1$, y por la segunda condición en (3.10), $r_f(p) = F$, lo cual es suficiente para la conclusión del teorema (como en el teorema 3.3.3).

3.3.2 Números de Mersenne

Un número de la forma $M_n = 2^n - 1$ es llamado *número de Mersenne*. Un problema teórico importante en la actualidad es el de determinar si existen infinitos números primos de Mersenne (llamados *primos de Mersenne*). Computacionalmente, se realizan muchos esfuerzos para hallar dichos números. Daremos un algoritmo bastante eficiente para calcular primos de Mersenne (un algoritmo polinomial), tanto así que los mayores primos conocidos son primos de Mersenne; el último, hallado a finales del año 2001, posee más de 4 millones de cifras. He aquí una tabla de los primos de Mersenne conocidos hasta el momento:

$2^2 - 1$	$2^3 - 1$	$2^5 - 1$	$2^7 - 1$	$2^{13} - 1$
$2^{17} - 1$	$2^{19} - 1$	$2^{31} - 1$	$2^{61} - 1$	$2^{89} - 1$
$2^{107} - 1$	$2^{127} - 1$	$2^{521} - 1$	$2^{607} - 1$	$2^{1279} - 1$
$2^{2203} - 1$	$2^{2281} - 1$	$2^{3217} - 1$	$2^{4253} - 1$	$2^{4423} - 1$
$2^{9869} - 1$	$2^{9941} - 1$	$2^{11213} - 1$	$2^{19937} - 1$	$2^{21701} - 1$
$2^{23209} - 1$	$2^{44497} - 1$	$2^{86243} - 1$	$2^{110503} - 1$	$2^{132049} - 1$
$2^{216091} - 1$	$2^{756839} - 1$	$2^{859433} - 1$	$2^{1257787} - 1$	$2^{1398269} - 1$
$2^{2976221} - 1$	$2^{3021377} - 1$	$2^{6972593} - 1$	$2^{13466917} - 1$	

¿Cómo determinar si un número de Mersenne es primo? Daremos un algoritmo polinomial para determinar esto. Pero antes debemos descartar algunos candidatos:

Teorema 3.3.5 Si $M_p = 2^p - 1$ es primo, entonces p es primo.

Prueba. En efecto, si $n = ab$ con $a, b > 1$ entonces $1 < 2^a - 1 < 2^n - 1$ y

$$2^n - 1 = 2^{ab} - 1 = (2^a)^b - 1 \equiv 1^b - 1 \equiv 0(\bmod 2^a - 1),$$

y por lo tanto M_n es compuesto.

Continuando con nuestro proceso de selección de candidatos, caracterizamos los factores primos de los números de Mersenne:

Teorema 3.3.6 (Euler) Sea $p > 2$ primo y q un divisor primo de M_p. Entonces $q \equiv 1(\bmod p)$ y $q \equiv \pm 1(\bmod 8)$.

Prueba. Si q divide a M_p entonces $2^p \equiv 1(\bmod q)$, de donde $\exp_q 2 = p$ (pues p es primo). Esto implica que $p|q-1$, es decir $q \equiv 1(\bmod p)$. Por otro lado,

$$2 \equiv 2^{p+1} \equiv (2^{(p+1)/2})^2 (\bmod q),$$

de donde $\left(\frac{2}{q}\right) = 1$, esto es, $q \equiv \pm 1(\bmod 8)$ (teorema 1.10.5).

Veamos ahora cómo verificar para ciertos valores especiales de p, algunos bastante grandes, que M_p no es primo:

Teorema 3.3.7 Sea p primo, $p \equiv 3(\bmod 4)$. Entonces $2p+1$ es primo si y sólo si $2p+1$ divide a M_p.

Prueba. Si $q = 2p+1$ es primo, entonces por el teorema 1.10.2

$$M_p = 2^p - 1 = 2^{(q-1)/2} - 1 \equiv \left(\frac{2}{q}\right) - 1 (\bmod q).$$

Además, $p \equiv 3(\bmod 4)$ implica que $q \equiv 7(\bmod 8)$, de donde $\left(\frac{2}{q}\right) = 1$ (teorema 1.10.5). Por lo tanto $M_p \equiv 0(\bmod q)$.

Recíprocamente, si $2p+1$ no fuese primo, entonces tendría un factor primo $r < p$ y luego $r \not\equiv 1(\bmod p)$. Si $2p+1|M_p$, entonces $r|M_p$ lo cual contradice la proposición anterior.

Los números primos p para los cuales $2p+1$ es primo son llamados primos de *Sophie Germain*; un ejemplo de tales primos es el número

$$p = 18458709 \cdot 2^{32611} - 1.$$

Volvamos ahora a nuestro problema inicial, el de determinar la primalidad de números de Mersenne. La siguiente proposición nos permite construir un algoritmo polinomial, el más utilizado en la práctica para determinar la primalidad de números de Mersenne:

Teorema 3.3.8 (Test de Lucas-Lehmer para primos de Mersenne) Consideremos la sucesión $(v_k)_{k\geq 0}$ definida por

$$v_0 = 4, \qquad v_{k+1} = v_k^2 - 2, \quad k = 0, 1, \ldots$$

Sea p un primo impar. Entonces $M_p = 2^p - 1$ es primo si y sólo si

$$v_{p-2} \equiv 0 (\bmod M_p).$$

Prueba. Sea $f(x) = x^2 - 4x + 1$, de donde $\Delta = 12$. Como

$$\begin{aligned} M_p &= 2^{p-2} \cdot 4 - 1 \equiv -1 \equiv 3 (\bmod 4), \\ M_p &= 2^p - 1 = 4^{(p-1)/2} \cdot 2 - 1 \equiv 1^{(p-1)/2} \cdot 2 - 1 \equiv 1 (\bmod 3), \end{aligned}$$

entonces

$$\begin{aligned} \left(\frac{\Delta}{M_p}\right) &= \left(\frac{2}{M_p}\right)^2 \left(\frac{3}{M_p}\right) = \left(\frac{3}{M_p}\right) \\ &= (-1)^{(\frac{M_p-1}{2})(\frac{3-1}{2})} \left(\frac{M_p}{3}\right) = (-1)\left(\frac{1}{3}\right) = -1. \end{aligned}$$

Aplicamos ahora el teorema anterior con

$$F = 2^{p-1} = \frac{M_p + 1}{2}.$$

Las condiciones (3.10) se reducen a la única condición

$$V_{2^{p-2}} \equiv 0 (\bmod M_p).$$

Pero

$$\begin{aligned} V_{2m} &\equiv x^{2m} + (4-x)^{2m} \equiv (x^m + (4-x)^m)^2 - 2x^m(4-x)^m \\ &= V_m^2 - 2 (\bmod (f(x), M_p)), \end{aligned}$$

pues $x(4-x) \equiv 1 (\bmod (f(x), M_p))$. Además, $v_1 = 4$. Por lo tanto, $V_{2^k} = v_k$, y del teorema anterior, si $v_{p-2} \equiv 0 (\bmod M_p)$, entonces M_p es primo.

Recíprocamente, supongamos que M_p es primo. Como $\left(\frac{\Delta}{M_p}\right) = -1$, entonces el anillo $\mathbb{Z}[x]/(f(x), M_p)$ es isomorfo al campo $F_{M_p^2}$. Por lo tanto (ver prueba del teorema 2.6.2) la aplicación $x \mapsto x^{M_p}$ es un automorfismo y

$$x^{M_p} \equiv 4 - x (\bmod (f(x), M_p)).$$

Ahora, como $(x-1)^2 \equiv 2x (\bmod (f(x), M_p))$ y por el teorema 1.10.2

$$2^{(M_p-1)/2} \equiv \left(\frac{2}{M_p}\right) \equiv 1 (\bmod M_p),$$

entonces

$$\begin{aligned}(x-1)^{M_p+1} &\equiv (2x)^{(M_p+1)/2} \equiv 2\cdot 2^{(M_p-1)/2}\cdot x^{(M_p+1)/2}\\ &\equiv 2x^{(M_p+1)/2}(\bmod (f(x), M_p)).\end{aligned}$$

Por otro lado

$$\begin{aligned}(x-1)^{M_p+1} &= (x-1)(x-1)^{M_p} \equiv (x-1)(x^{M_p}-1 \equiv (x-1)(3-x)\\ &\equiv -2(\bmod (f(x), M_p)).\end{aligned}$$

Por lo tanto, $x^{(M_p+1)/2} \equiv -1(\bmod (f(x), M_p))$, esto es,

$$x^{2^{p-1}} \equiv -1(\bmod (f(x), M_p)).$$

Utilizando el automorfismo $x \mapsto x^{M_p}$, tenemos

$$(4-x)^{2^{p-1}} \equiv -1(\bmod (f(x), M_p)),$$

de modo que

$$U_{2^{p-1}} \equiv 0(\bmod M_p).$$

Si $U_{2^{p-2}} \equiv 0(\bmod M_p)$, entonces

$$x^{2^{p-2}} \equiv (4-x)^{2^{p-2}}(\bmod (f(x), M_p))$$

y luego

$$-1 \equiv x^{2^{p-1}} \equiv x^{2^{p-2}}(4-x)^{2^{p-2}} \equiv (x(4-x))^{2^{p-2}} \equiv 1^{2^{p-2}} \equiv 1(\bmod (f(x), M_p)),$$

absurdo. Como $U_{2^{p-1}} = U_{2^{p-2}}V_{2^{p-2}}$, tenemos que $V_{2^{p-2}} \equiv 0(\bmod M_p)$; pero $V_{2^{p-2}} = v_{p-2}$. Esto concluye la prueba.

Algoritmo 3.3.1 (Test de Lucas-Lehmer para primos de Mersenne) Este algoritmo verifica la primalidad de un número de Mersenne $M_p = 2^p - 1$.

Entrada:	p primo impar
$v = 4$	
para $k = 1, \ldots, p-2$	
	$v = (v^2 - 2) \bmod (2^p - 1)$
Salida:	si $v = 0$ entonces "es primo"
	sino "es compuesto"

El número de iteraciones de este algoritmo es $p - 2 = O(\log M_p)$, cada una de costo $O(\log^2 M_p)$. Por lo tanto la complejidad de este algoritmo es $O(\log^3 M_p)$, tiempo polinomial.

Capítulo 4

El algoritmo de Agrawal-Kayal-Saxena

Este algoritmo, publicado en agosto del 2002 por Manindra Agrawal, Neeraj Kayal y Nitin Saxena [22], es el primer algoritmo que permite determinar, en tiempo polinomial, si un número cualquiera es primo o compuesto; esto sin depender de ninguna hipótesis sin demostrar.

4.1 El algoritmo AKS

El método se basa en el siguiente teorema:

Teorema 4.1.1 Sean a, n enteros con n impar, $(a, n) = 1$. Entonces n es primo si y sólo si

$$(x-a)^n \equiv (x^n - a)(\bmod n) \tag{4.1}$$

(la congruencia es en $\mathbb{Z}_n[x]$).

Prueba. Sea n primo. Para $0 < i < n$, el coeficiente de x^i en la expansión de $(x-a)^n$ es $\binom{n}{i}(-a)^{n-i}$. Como

$$\binom{n}{i} = \frac{n}{i}\binom{n-1}{i-1}, \quad 0 < i < n$$

y $(i, n) = 1$, entonces $n \mid \binom{n}{i}$. Por lo tanto todos mencionados coeficientes son congruentes a $0 \mod n$, y

$$(x-a)^n \equiv x^n - a^n \equiv x^n - a(\bmod n).$$

Si n es compuesto, sea $q \mid n$, q primo y sea $q^k \| n$. Veamos que $q^k \nmid \binom{n}{q}$: en efecto,

$$\binom{n}{q} = \frac{n!}{(n-q)!q!} = \frac{(n-q+1)(n-q+2)\ldots(n-1)(n/q)}{(q-1)!};$$

ahora observe que $q^{k-1} \| n/q$ y $q \nmid (n-q+i)$ para $0 < i < n$, y además $((q-1)!, q) = 1$. Por lo tanto $q^{k-1} \| \binom{n}{q}$ (y de ahí $q^k \nmid \binom{n}{q}$). Ahora como $(a, q) = 1$, entonces $(a^{n-q}, q^k) = 1$ y luego $\binom{n}{q} a^{n-q} \equiv 0 (\bmod n)$; pues de otro modo $\binom{n}{q} a^{n-q} \equiv 0 (\bmod q^k)$ y luego $q^k \| \binom{n}{q}$, absurdo. Por lo tanto el coeficiente de x^q en la expansión de $(x-a)^n$ no es cero $\bmod n$, $0 < q < n$ y por lo tanto no se cumple (4.1).

Si bien la identidad (4.1) formará la base del método para probar primalidad, intentar verificarla directamente tomaría tiempo exponencial (puesto que hay que calcular p coeficientes de $(x-a)^p$). En vez de ello, verificamos la congruencia

$$(x-a)^n \equiv x^n - a (\bmod x^r - 1, n). \tag{4.2}$$

para un polinomio dado $x^r - 1$ en $\mathbb{Z}_p[x]$. Veremos que verificar esta congruencia es mucho más rápido que la identidad (4.1); de hecho toma tiempo polinomial. Es evidente de lo anterior que todo primo cumple (4.2), aunque algunos enteros compuestos pueden también verificar la congruencia. Para descartar esta posibilidad debemos escoger adecuadamente r.

Presentamos a continuación el algoritmo AKS; aquí $P(m)$ denota el mayor factor primo del entero m. Y en adelante denotaremos por log al logaritmo en base 2, $\log_2$.

Algoritmo 4.1.1 Dado un entero impar $n > 1$, este algoritmo determina si el número es primo o compuesto.

```
Entrada: n > 1 entero impar
primo = Verdadero
m = ⌊log n⌋
k = 2
mientras (k ≤ m) ∧ primo
    [verificando si n = a^k]
    a = 0
    i = m
    mientras i ≥ 0 ∧ primo
        si (a + 2^i)^k ≤ n
            a = a + 2^i
        si a^k = n
            primo = Falso
        i = i − 1
    k = k + 1
iterar = primo
r = 1
mientras (r < n) ∧ iterar
    r = r + 1
    si (n, r) ≠ 1
        primo = Falso
        iterar = Falso
    sino
        si (r es primo)
            q = P(r − 1)
            si (q ≥ 4√r log n ∧ n^((r−1)/q) ≠ 1(mod r))
                iterar = Falso
si primo
    a = 1
    mientras (a ≤ 2√r log n ∧ primo)
        si ((x − a)^n ≠ (x^n − a)(mod x^r − 1, n))
            primo = Falso
        a = a + 1
Salida: la respuesta a ¿es n primo? en primo
```

Nos hacemos entonces una pregunta importante: si estamos intentando probar que un número es primo, ¿porqué en un cierto paso del algoritmo se nos pide probar que r

es primo, y cómo hacemos eso? La buena noticia, que veremos luego, es que r será lo suficientemente pequeño como para el método de división por prueba de la sección 2.1 para determinar si es primo o no (de hecho, r es $O(polinomio(\log n))$, como veremos más adelante); es igual de fácil calcular $P(r-1)$.

Ahora vamos a demostrar que el algoritmo, efectivamente, detecta si un número dado es primo o compuesto. Necesitamos del siguiente lema.

Lema 4.1.1 Sean p, r primos distintos.

(i) Sea $f(x)$ un polinomio con coeficientes integrales, entonces

$$f(x)^p \equiv f(x^p) (\bmod p).$$

(ii) Sea $h(x)$ un factor de $x^r - 1$. Si $m \equiv m' (\bmod r)$, entonces

$$x^m \equiv x^{m'} (\bmod h(x)).$$

(iii) En $\mathbb{F}_p$, $\dfrac{x^r-1}{x-1}$ se factoriza en polinomios irreducibles de grado $\exp_r(p)$.

Prueba.

(i) Sean $f(x)$, $g(x)$ polinomios entonces

$$(f(x)+g(x))^p = \sum_{i=0}^{p} \binom{p}{i} f(x)^i g(x)^{p-i},$$

y para $0 < i < p$, $p | \binom{p}{i}$ y por lo tanto

$$(f(x)+g(x))^p \equiv f(x)^p + g(x)^p (\bmod p).$$

Aplicando esto a $f(x) = a_0 + a_1 x + \cdots + a_d x^d$, y dado que $a_i^p \equiv a_i (\bmod p)$ para todo i,

$$f(x)^p \equiv a_0 + a_1 x^p + \cdots + a_d (x^d)^p \equiv f(x^p) (\bmod p).$$

(ii) Considerando que $m = m' + kr$, y $x^r \equiv 1 (\bmod h(x))$, entonces

$$x^m \equiv x^{m'} \cdot (x^r)^k \equiv x^{m'} \cdot 1^k \equiv x^{m'} (\bmod h(x)).$$

(iii) Sean $d = \exp_r(p)$; $q(x) = \dfrac{x^r-1}{x-1}$. Entonces $p^d \equiv 1 (\bmod r)$ y por lo anterior

$$x^{p^d} \equiv x (\bmod h(x)). \tag{4.3}$$

Sea $h(x)$ un factor irreducible de $q(x)$ en $\mathbb{F}_p[x]$, y r el grado de $h(x)$. El anillo $A = \mathbb{F}_p[x]/h(x)$ es un campo con p^k elementos y el subgrupo multiplicativo de A es cíclico con un generador $g(x)$. Además, de (ii) y (4.3)

$$g(x)^{p^d} \equiv g(x^{p^d}) \equiv g(x)(\bmod h(x), p),$$

y de ahí $g(x)^{p^d-1} \equiv 1(\bmod h(x), p)$. Ahora como $p^k - 1 = o_{A^*}(g(x))$ (orden en A^*) entonces $(p^k - 1)|(p^d - 1)$, esto es

$$p^d \equiv 1(\bmod p^k - 1);$$

y como $\exp_{p^k-1}(p) = k$ trivialmente, entonces $k|d$.

Ahora, $x^r \equiv 1(\bmod h(x), p)$ y $x \not\equiv 1(\bmod h(x), p)$, de donde como r es primo, $o_{A^*}(x) = r$. Como también

$$x^{p^k-1} \equiv 1(\bmod h(x), p),$$

concluimos que $r|p^k - 1$, esto es, $p^k \equiv 1(\bmod r)$ de donde $d|k$. Concluimos que $k = d$.

Un recorrido simple del algoritmo revela que un número primo pasa el test AKS sin problemas. En efecto, si n es primo, es imposible que sea $n = a^k$ para algún entero a y $k \geq 2$; esto hace que n pase el primer bloque del algoritmo. Para $r < n$, siempre $(n, r) = 1$ y el segundo bloque del algoritmo determinará algún r (inclusive $r = n$). Ya vimos que n pasará el tercer bloque del algoritmo ya que satisface (4.2) para cualquier r.

Ahora debemos probar que un número compuesto no pasará el test AKS. Supongamos que un número compuesto n pasa el algoritmo como primo, en lo que queda de la sección.

El primer bloque del algoritmo determina si n es una potencia exacta, esto es, si $n = a^k$ para $a \geq 2$, $k \geq 2$ enteros. Veamos que si $n = a^k$, entonces $n = a^k \geq 2^k$ y luego $k \leq \log n$, y luego $k \leq \lfloor \log n \rfloor$. Esto restringe la búsqueda de los exponentes k. A continuación calculamos $a = \lfloor n^{1/k} \rfloor$ por búsqueda binaria: en efecto, el algoritmo en cada paso reconstruye la representación en base 2 del mayor entero a tal que $a^k \leq n$ (observe que $a \leq a^k \leq n$ así que hacemos la búsqueda entre los números $\leq n$). Como

$$\lfloor n^{1/k} \rfloor^k \leq (n^{1/k})^k = n \leq (\lfloor n^{1/k} \rfloor + 1)^k$$

así que el mayor entero a con $a^k \leq n$ es efectivamente $a = \lfloor n^{1/k} \rfloor$. Si $n = a^k$ entonces n es compuesto y el algoritmo finaliza sin ejecutar los siguientes bloques principales.

Analicemos el segundo bloque del algoritmo. Como n es compuesto, $(n, r) \neq 1$ para algún $r < n$ y la única forma en que n pase el segundo bloque del algoritmo es hallando

r con las condiciones requeridas. Supongamos entonces que tenemos r hallado por el algoritmo. Escribimos $n = p_1 p_2 \cdots p_k$ con p_i primos (no necesariamente distintos), $1 \leq i \leq k$. Sea $f = \text{mcm}\{\exp_r(p_i)\}_{i=1,\ldots,k}$; como $\exp_r(p_i)|f$ para cada i,

$$p_i^f \equiv 1 (\bmod r)$$

y luego

$$n^f = p_1^f p_2^f \cdots p_k^f \equiv 1 (\bmod r),$$

y entonces $\exp_r(n)|f$. Sea ahora $q = P(r-1)$. Como $(n, r) = 1$,

$$n^{r-1} \equiv 1 (\bmod r) \quad \text{y} \quad n^{\frac{r-1}{q}} \not\equiv 1 (\bmod r),$$

y entonces

$$\exp_r(n) | r-1 \text{ pero } \exp_r(n) \not| \frac{r-1}{q}$$

y por lo tanto $q|\exp_r(n)$. De lo anterior, $q|\text{mcm}\{\exp_r(p_i)\}_{i=1,\ldots,k}$ y existe entonces un primo $p|n$ con $q|\exp_r(p)$. Fijemos tal factor p, de ahora en adelante.

Sea $l = \lfloor 2\sqrt{r} \log n \rfloor$. Sea $h(x)$ un factor de $x^r - 1$ de grado $d = \exp_r(p)$ irreducible en $\mathbb{F}_p$, obtenido del lema 4.1.1, (iii).

Lema 4.1.2 En el campo $\mathbb{F}_p[x]/(h(x))$, el grupo generado por los binomios $(x-a)$, $1 \leq a \leq l$, i.e.

$$G = \left\{ \prod_{1 \leq a \leq l} (x-a)^{\alpha_a} : \alpha_a \geq 0,\ 1 \leq a \leq l \right\}$$

es cíclico y de orden $|G| > \left(\frac{d}{l}\right)^l$.

Prueba. Como G es un subgrupo del grupo cíclico $(\mathbb{F}_p[x]/(h(x)))^*$, también es cíclico (teorema 1.9.2). Consideremos el conjunto

$$S = \left\{ \prod_{1 \leq a \leq l} (x-a)^{\alpha_a} : \sum_{1 \leq a \leq l} \alpha_a \leq d-1,\ \alpha_a \geq 0,\ 1 \leq a \leq l \right\}.$$

Probaremos que los elementos de S son distintos en $\mathbb{F}_p[x]/(h(x))$. Observe que

$$r > q \geq 4\sqrt{r} \log n > l.$$

Si algunos de los enteros a con $1 \leq a \leq l$ fuesen congruentes módulo p, i.e.

$$a_1 \equiv a_2 (\bmod p) \text{ con } a_1 < a_2$$

entonces $p|(a_2 - a_1)$ implicaría

$$p \leq a_2 - a_1 < l < r$$

y el segundo bloque del algoritmo habría identificado a n como compuesto (al evaluar $(n, p) = p$), antes de hallar r, absurdo.

Como los enteros a son incongruentes módulo p, entonces los elementos de S son distintos módulo p, y por lo tanto en $\mathbb{F}_p[x]/(h(x))$, ya que cada uno de ellos tiene grado $< d$ (el grado de $h(x)$). Ahora

$$\#S = \binom{l+d-1}{l} = \frac{(l+d-1)(l+d-2)\cdots(d)}{l!} > \left(\frac{d}{l}\right)^l,$$

y como $S \subset G$, $|G| > \left(\frac{d}{l}\right)^l$ como queríamos.

Recordemos ahora que $q|\exp_r(p) = d$; de donde

$$d \geq q \geq 4\sqrt{r}\log n \geq 2\lfloor 2\sqrt{r}\log n \rfloor = 2l;$$

luego $|G| > 2^l > 2^{2\sqrt{r}\log n - 1} = \frac{n^{2\sqrt{n}}}{2}$.

Sea $g(x)$ un generador de G; entonces el orden de G en $(\mathbb{F}_p[x]/(h(x)))^*$ cumple

$$o(g) > n^{2\sqrt{n}}/2.$$

Definimos

$$I_g = \{m : g(x)^m \equiv g(x^m)(\bmod x^r - 1, p)\}.$$

Lema 4.1.3 El conjunto I_g es cerrado con respecto a la multiplicación.

Prueba. Sean $m, m' \in I_g$. Como $m' \in I_g$,

$$g(x)^{m'} \equiv g(x^{m'})(\bmod x^r - 1, p);$$

reemplazando x por x^m obtenemos

$$g(x^m)^{m'} \equiv g(x^{mm'})(\bmod x^{mr} - 1, p),$$

pero $x^r - 1 | x^{mr-1} - 1$, entonces

$$g(x^m)^{m'} \equiv g(x^{mm'})(\bmod x^r - 1, p).$$

Por esto y como $m \in I_g$,

$$g(x)^{mm'} \equiv (g(x)^m)^{m'} \equiv g(x^m)^{m'} \equiv g(x^{mm'})(\bmod x^r - 1, p).$$

Lema 4.1.4 Para $m, m' \in I_g$, si

$$m \equiv m' (\bmod r), \text{ entonces } m \equiv m' (\bmod o(g)).$$

Prueba. Escribimos $m' = m + kr$ para algún $k \geq 0$. Del lema 4.1.1, (ii),

$$x^{m'} \equiv x^m (\bmod h(x), p).$$

Luego

$$g(x)^m g(x)^{kr} = g(x)^{m+kr} = g(x)^{m'} \equiv g(x^{m'}) \equiv g(x^m) \equiv g(x)^m (\bmod h(x), p).$$

Como $g(x) \not\equiv 0$, entonces $g(x)^m \not\equiv 0$ y por lo tanto

$$g(x)^{kr} \equiv 1 (\bmod h(x), p);$$

de donde $kr \equiv 0 (\bmod o(g))$ y $m \equiv m' (\bmod o(g))$.

A continuación presentamos el argumento final de la prueba de que un entero compuesto no puede pasar el algoritmo como primo. Si n supera el tercer bloque principal del algoritmo,

$$(x - a)^n \equiv (x^n - a) (\bmod x^r - 1, p), \quad \text{para } 1 \leq a \leq l. \tag{4.4}$$

Como $g(x) \in G$, $g(x)$ es producto de elementos que cumplen (4.4), luego

$$g(x)^n \equiv g(x^n) (\bmod x^r - 1, p)$$

y por lo tanto $n \in I_g$. También tenemos que $p \in I_g$ por el lema 4.1.1 (i) y, trivialmente, $1 \in I_g$.

Sea

$$E = \{n^i p^j : 0 \leq i, j \leq \lfloor \sqrt{r} \rfloor\}.$$

Por el lema 4.1.3, $E \subset I_g$. Como

$$\#E = (1 + \lfloor \sqrt{r} \rfloor)^2 > r,$$

existen elementos $n^i p^j$ y $n^{i'} p^{j'}$ en E con $i \neq i'$ o $j \neq j'$ y

$$n^i p^j \equiv n^{i'} p^{j'} (\bmod r).$$

Por el lema 4.1.4, $n^i p^j \equiv n^{i'} p^{j'} (\bmod o(g))$, o equivalentemente

$$n^{i-i'} \equiv p^{j'-j} (\bmod o(g)). \tag{4.5}$$

Recordemos ahora que $o(g) > n^{2\sqrt{r}}/2$. Veamos que

$$n^{|i-i'|} \leq n^{\lfloor\sqrt{r}\rfloor} \leq n^{\sqrt{r}};$$

además como $p < n$ y $p|n$, entonces $n/p \geq 2$ y

$$p^{|j-j'|} \leq p^{\lfloor\sqrt{r}\rfloor} \leq \left(\frac{n}{2}\right)^{\lfloor\sqrt{r}\rfloor} = \frac{n^{\lfloor\sqrt{r}\rfloor}}{2^{\lfloor\sqrt{r}\rfloor}} \leq \frac{n^{\sqrt{r}}}{2}$$

y luego la congruencia (4.5) ha de ser una igualdad. Como p es primo y por la unicidad de la descomposición de n en factores primos, $n = p^k$ y $k \geq 2$ pues n es compuesto. Esto es absurdo pues el primer bloque del algoritmo detecta potencias exactas de enteros.

4.2 Análisis de complejidad del algoritmo AKS

Dedicaremos esta sección a medir la velocidad del algoritmo AKS. Un vistazo simple al algoritmo revela que la velocidad depende de cuán rápido se encuentre el elemento r; puesto que el segundo bloque del algoritmo realiza r iteraciones. Veremos que r es $O(polinomio(\log n))$, lo cual nos dará un algoritmo polinomial.

Necesitaremos el siguiente hecho de la teoría analítica de números, correspondiente a la *teoría de cribas*; la demostración de este hecho escapa a los métodos utilizados en este trabajo. Para ver la prueba y extensiones de este hecho, ver [4], [15].

Lema 4.2.1 Existen constantes $c > 0$ y n_0, tales que para $x \geq n_0$,

$$\#\{p : p \text{ primo }, p \leq x \text{ y } P(p-1) > x^{2/3}\} \geq \frac{cx}{\log x}.$$

Teorema 4.2.1 Existen constantes positivas c_1 y c_2 con $c_1 < c_2$ para las cuales existe un primo r en el intervalo $[c_1(\log n)^6, c_2(\log n)^6]$ tal que

$$q = P(r-1) \text{ cumple } q \geq 4\sqrt{r}\log n \text{ y } n^{\frac{r-1}{q}} \not\equiv 1 (\bmod r).$$

Prueba. Llamaremos a un primo r cumpliendo $P(r-1) > x^{2/3}$ por *primo especial*; denotaremos el número de tales primos en el intervalo $[x, y]$ por $\pi_{SP}[x, y]$. Por el lema anterior y el teorema 1.13.1, para n suficientemente grande,

$$\begin{aligned} \pi_{SP}[c_1(\log n)^6, c_2(\log n)^6] &\geq \pi_{SP}[1, c_2(\log n)^6] - \pi(c_1(\log n)^6) \\ &\geq \frac{cc_2(\log n)^6}{7\log\log n} - \frac{5c_1(\log n)^6}{6\log\log n} = \frac{(\log n)^6}{\log\log n}\left(\frac{cc_2}{7} - \frac{5c_1}{6}\right), \end{aligned}$$

donde hemos escogido previamente $c_1 \geq 4^6$ y c_2 lo suficientemente grande para que $c_3 = \frac{cc_2}{7} - \frac{5c_1}{6}$ sea positivo.

Sea $x = c_2(\log n)^6$. Consideremos el producto

$$Q = \prod_{k \leq x^{1/3}} (n^k - 1).$$

Veamos que Q tiene a lo más $x^{2/3} \log n$ factores primos: en efecto, si

$$Q = p_1 p_2 \cdots p_s, \quad \text{entonces} \quad Q \geq 2^s$$

y luego

$$\begin{aligned} s &\leq \log Q = \sum_{k \leq x^{1/3}} \log(n^k - 1) < \sum_{k \leq x^{1/3}} \log(n^k) = \sum_{k \leq \lfloor x^{1/3} \rfloor} k \log n \\ &= \frac{\lfloor x^{1/3} \rfloor (\lfloor x^{1/3} \rfloor + 1) \log n}{2} \leq x^{2/3} \log n. \end{aligned}$$

Note ahora que

$$x^{2/3} \log n < \frac{c_3 (\log n)^6}{\log \log n},$$

esto implica que el número de factores de Q es estrictamente menor al número de primos especiales en $[c_1(\log n)^6, c_2(\log n)^6]$. Por lo tanto, existe un primo especial $r \leq x$ (en el intervalo) que no divide a Q y tal que $q = P(r-1)$ cumple

$$q > x^{2/3} \geq x^{1/6} x^{1/2} \geq 4 \log n \sqrt{r}.$$

Como $q \not| Q$, entonces $n^i \not\equiv 1 (\bmod r)$, para todo $i \leq x^{1/3}$. Ahora $q > r^{2/3}$, entonces $\dfrac{r-1}{q} < r^{1/3} \leq x^{1/3}$ y por lo tanto

$$n^{\frac{r-1}{q}} \not\equiv 1 (\bmod r).$$

Ahora estamos listos para medir la velocidad del algoritmo AKS.

Teorema 4.2.2 La complejidad asintótica del algoritmo es $\tilde{O}(\log^{19} n)$.

Midamos la complejidad del primer bloque del algoritmo: Se realizan $k = O(\log n)$ iteraciones, para determinar si $n = a^k$ se realizan a su vez $m = O(\log n)$ iteraciones en las cuales la operación más costosa (cálculo de a^k) toma, utilizando una variación del algoritmo 1.5.1, tiempo $O(\log k (k \log n)^2) = O((\log^4 n) \log \log n)$; luego la complejidad total es $O(\log^6 n \log \log n) = \tilde{O}(\log^6 n)$.

En el segundo bloque del algoritmo, se realizan $r = O(\log^6 n)$ iteraciones: primero se realiza el cálculo de (n, r) que tiene costo $O(\log n \, polinomio(\log \log n))$; determinar si r es primo por el método de división por prueba tiene complejidad $O(r^{1/2}(\log r)^2) = O(r^{1/2} polinomio(\log \log n))$ (lo mismo para calcular q); y los siguientes pasos tienen costo

$O(polinomio(\log\log n))$. Así cada iteración tiene costo $\tilde{O}(r^{1/2}) = \tilde{O}(\log^3 n)$ y por lo tanto el segundo bloque del algoritmo tiene complejidad total $\tilde{O}(\log^9 n)$.

El tercer bloque del algoritmo realiza operaciones modulares sobre polinomios; en total $O(r^{1/2}\log n) = O(\log^4 n)$ iteraciones. Veamos que el costo de cada iteración es $O(r^2\log^3 n) = O(\log^{15} n)$. En efecto, dado que $x^r \equiv 1$ en $R = \mathbb{Z}_n[x]/(x^r - 1)$, un sistema de representantes de R es

$$S = \{a_0 + a_1 x + a_2 x^2 + \cdots + a_{r-1}x^{r-1} : 0 \leq a_j < n\}.$$

La multiplicación en S será la multiplicación usual de polinomios en $\mathbb{Z}_n[x]$, adjuntando la regla de reemplazo $x^r = 1$; esto requiere $O(r^2)$ operaciones entre sumas y productos, en $\mathbb{Z}_n$; cada una con costo $O(\log^2 n)$. Con una variación del método de exponenciación en $\mathbb{Z}_n$ (algoritmo 1.5.1), podemos evaluar rápidamente $(x-a)^n$; recordemos que el método requería $O(\log n)$ iteraciones; que acabamos de ver que tienen costo $O(r^2\log^2 n)$. De aquí se sigue el resultado.

4.2.1 Primos de Sophie Germain

Recordemos que un primo de Sophie Germain es un primo r tal que $\frac{r-1}{2}$ también es primo; tales r y $\frac{r-1}{2}$ son llamados *coprimos de Sophie Germain*. Observe entonces que, para un primo de Sophie Germain r, $P(r-1) = \frac{r-1}{2}$. Esta es una propiedad más fuerte que la que buscamos para los elementos r en el algoritmo AKS.

Una conjetura sobre la densidad de los primos de Sophie Germain bastante conocida nos permitiría (de ser probada) mejorar la cota superior de la velocidad del algoritmo:

Conjetura 4.2.1 El número de coprimos de Sophie Germain es asintóticamente $\frac{Dx}{\log^2 x}$, para alguna constante D.

Con esto, tenemos el siguiente estimado:

Teorema 4.2.3 Asumiendo la conjetura 4.2.1, existe un r cumpliendo las condiciones del teorema 4.2.1 en el intervalo $[64\log^2 n, c_2\log^2 n]$ para $n > n_0$, donde n_0 y c_2 son constantes positivas.

Prueba. Si $r, q = \frac{r-1}{2}$ son primos, entonces

$$\exp_r(n) \in \{1, 2, q, 2q = r-1\}.$$

Ahora si $\exp_r(n) = d$, con $d = 1$ o $d = 2$, entonces

$$n^d \equiv 1 (\bmod r) \text{ i.e. } r | n^d - 1$$

y de ahí $r|n^2 - 1$. Como el número de factores de $n^2 - 1$ es $\leq \log(n^2 - 1) < 2 \log n$, entonces hay a lo más $2 \log n$ primos para los cuales $\exp_r(n) \in \{1, 2\}$.

Consideremos entonces los otros coprimos de Sophie Germain, aquellos para los cuales $\exp_r(n) \geq \dfrac{r-1}{2}$. Imponiendo la condición

$$q = \frac{r-1}{2} \geq 4\sqrt{r} \log n,$$

obtenemos la condición $r \geq 64 \log^2 n$.

Ahora tomemos $r \in [64 \log^2 n, c_2 \log^2 n]$, c_2 por determinar. Por la conjetura, el número de coprimos de Sophie Germain menores a $c_2 \log^2 n$ es asintóticamente $\dfrac{Dc_2 \log^2 n}{\log^2(c_2 \log^2 n)}$. De aquellos, a lo más $\dfrac{D64 \log^2 n}{\log^2(64 \log^2 n)}$ son menores que $64 \log^2 n$. De los que quedan, descartamos los primos para los cuales el orden de n módulo r es 1 o 2, que son a lo más $2 \log n$. Así que escogemos c_2 tal que

$$\frac{Dc_2 \log^2 n}{\log^2(c_2 \log^2 n)} > \frac{D64 \log^2 n}{\log^2(64 \log^2 n)} + 2 \log n;$$

esto es $c_2 > 100$ (y n suficientemente grande).

Gracias a este estimado, la complejidad heurística del algoritmo es $\tilde{O}(\log^9 n)$.

Capítulo 5

Criptografía RSA

En este capítulo presentamos el método de criptografía de clave pública más utilizado en Internet: el RSA. Las ideas básicas, que llevaron a la construcción del método, fueron dadas por Diffie y Hellman en 1976 (ver [1]); la implementación efectiva del método fue dada por Rivest, Shamir y Adleman en 1978 (y por ello llamado RSA). Los algoritmos estudiados en los capítulos anteriores servirán para obtener números primos grandes (de 100 cifras o más), lo que son necesarios para obtener las claves (de codificación y decodificación) con las cuales trabaja el método.

5.1 Encriptación Asimétrica.

También conocida como *criptografía de clave pública*, esta forma de encriptación utiliza dos claves: Una clave pública que puede ser libremente distribuida y una clave privada que es mantenida en secreto. Veamos cómo funciona:
Supongamos que un usuario dado del sistema criptográfico quiere enviar y recibir mensajes seguros. El usuario hace público, en una lista (algo así como el directorio telefónico) un método C para que otros usuarios codifiquen los mensajes que le serán enviados, y mantiene en secreto un procedimiento D para decodificarlos; con las siguientes propiedades:

(i) $D(C(M)) = M$ para cualquier mensaje dado M (como era de esperarse);

(ii) $C(D(M)) = M$ para cualquier mensaje M. Observamos que podemos aplicar el procedimiento de decodificado D a un mensaje que *no* ha sido codificado;

(iii) C y D pueden calcularse rápidamente;

(iv) Es *imposible* calcular D tan sólo a partir de C (a pesar de que las claves están matemáticamente relacionadas).

Suponga que un usuario 1 del sistema (con procedimientos C_1 para codificar y D_1 para decodificar) desea enviar un mensaje M a un usuario 2 (con procedimientos C_2, D_2). Entonces 1 envía a 2 el mensaje

$$M' = C_2(D_1(M)).$$

Observe que 1 conoce C_2 pues es público. Entonces al recibir el usuario 2 el mensaje M', aplica D_2 (que sólo lo conoce él) y luego C_1 (que es público), obteniendo de vuelta

$$M = C_1(D_2(M')).$$

De esta manera el usuario 1 se asegura que 2 es el único que puede leer su mensaje y 2 se asegura que 1 es el verdadero remitente del mismo (de otro modo aplicar C_1 no debería recuperar el mensaje).

Lo más importante es el hecho de que dos usuarios del sistema, sin haber tenido algún contacto previo, pueden intercambiar mensajes seguros. Esto es lo que hace al método ideal para su uso en una red pública como Internet. Los navegadores más utilizados actualmente, *Explorer* y *Netscape*, utilizan el RSA.

5.2 El método

Ahora veremos cómo funciona el método. Para cifrar un mensaje, debemos dividirlo en bloques, digamos de s letras cada uno, que serán codificados en bloques de t letras (donde $s < t$). Nuestro mensaje está escrito en un alfabeto $\mathcal{A}$ de L letras.
Cada usuario escoge aleatoriamente un número entero n tal que

$$L^s < n < L^t$$

y enteros e, d tales que

$$a^{ed} \equiv a (\mathrm{mod}\, n), \text{ para } 0 \leq a < n.$$

Dado un bloque B de mensaje original,

$$B = [l_1 l_2 \cdots l_s], \quad l_i \in \mathcal{A}, 1 \leq i \leq s.$$

B puede tener L^s formas distintas. Como $L^s < n$, B puede ser visto como un entero a en $\mathbb{Z}_n$. Calculamos

$$a' \equiv a^e (\mathrm{mod}\, n),$$

de modo que $0 \leq a' < n$. Ahora $n < L^t$ y a' puede ser visto como un bloque B' de t letras en el alfabeto $\mathcal{A}$. Así codificamos B en B'. La unión de estos bloques nos da el mensaje

codificado.
Para decodificar un mensaje, simplemente dividimos el mensaje en bloques B' de t letras, los retornamos a su representación en la forma a' de $\mathbb{Z}_n$ y calculamos

$$(a')^d \equiv a^{ed} (\mathrm{mod}\, n) \equiv a (\mathrm{mod}\, n),$$

obteniendo de vuelta B.
En resumidas cuentas, tenemos:

- Información pre-acordada (para todos los usuarios): L, s, t. Estos parámetros determinan el tamaño de la clave.
- Clave pública del usuario: (n, e).
- Clave privada: d.

Luego la seguridad del método RSA se encuentra en la imposibilidad de hallar d conociendo tan sólo n y e. Justificaremos esto más adelante.

¿Cómo escoger n, e y d? Buscamos un par de números primos distintos p y q tales que

$$L^s < n = pq < L^t.$$

Escogemos aleatoriamente e con $0 \leq e < \varphi(n) = (p-1)(q-1)$ y $(e, \varphi(n)) = 1$. d podrá ser entonces el inverso de e en $\mathbb{Z}^*_{\varphi(n)}$, pues tenemos el siguiente

Teorema 5.2.1 Sea $n = pq$, donde p y q son primos distintos. Si $f \equiv 1 (\mathrm{mod}\, \varphi(n))$ entonces

$$a^f \equiv a (\mathrm{mod}\, n),$$

para todo $0 \leq a < n$.

Prueba. En efecto, $f \equiv 1 (\mathrm{mod}\, \varphi(n))$ implica

$$f = \lambda\varphi(n) + 1$$

con $\lambda \in \mathbb{Z}$. Como $f = \lambda\varphi(p)\varphi(q) + 1$, si $(a, p) = 1$ por el pequeño teorema de Fermat:

$$a^f \equiv (a^{\varphi(p)})^{\lambda\varphi(q)} \cdot a \equiv 1^{\lambda\varphi(q)} \cdot a \equiv a (\mathrm{mod}\, p),$$

y si $(a, p) \neq 1$, entonces $(a, p) = p$ y

$$a^f \equiv 0 (\mathrm{mod}\, p) \equiv a (\mathrm{mod}\, p).$$

Así

$$a^f \equiv a (\mathrm{mod}\, p),$$

para cualquier a. Análogamente

$$a^f \equiv a(\bmod q).$$

Como $n = pq$, y p, q son primos distintos entonces

$$a^f \equiv a(\bmod n)$$

como queríamos.

Evidentemente, para que el sistema sea seguro, en la práctica, necesitamos que p y q sean primos relativamente grandes (digamos de 100 o 200 cifras), de modo que sea difícil calcular d. Es aquí donde se utilizan los métodos estudiados en los capítulos anteriores, y por ello la necesidad de eficiencia de los mismos.

5.3 Seguridad del RSA

Ahora mostraremos cómo es que la seguridad del método RSA está ligada a la dificultad de factorizar $n = pq$. En un intento por *romper el código* (esto es, permitir la lectura de un mensaje a un usuario ilegítimo), presentamos las siguientes alternativas:

Encontrar $\varphi(n)$

Supongamos que queremos hallar $\varphi(n)$ para calcular e. Como $n = pq$, $\varphi(n) = (p-1)(q-1)$, entonces p y q son raíces de

$$x^2 + (\varphi(n) - n - 1)x + n = 0.$$

Si $\Delta^2 = (\varphi(n) - n - 1)^2 - 4n$, entonces

$$p, q = \frac{-(\varphi(n) - n - 1) \pm \Delta}{2}.$$

Ahora veamos lo rápido que se puede calcular Δ:

Lema 5.3.1 El costo de calcular $\lfloor \sqrt{n} \rfloor$ es $O(\log^3 n)$.

Para esto presentamos el siguiente algoritmo:

Algoritmo 5.3.1 Este algoritmo calcula $\lfloor \sqrt{n} \rfloor$ por un método de aproximación.

Entrada: entero n
$a = 0$ $b = n$
$x' = \lfloor (a+b)/2 \rfloor$
mientras $((b-a) \neq 1 \wedge g(x') \neq 0)$
si $g(x') > 0$
$b = x'$
si $g(x') < 0$
$a = x'$
Salida: $\lfloor \sqrt{n} \rfloor = x'$

Calculemos el costo del algoritmo: Cada iteración tiene un costo igual a $O(\log^2 n)$, con un total de $O(\log n)$ iteraciones, ya que en cada paso el tamaño del problema (el intervalo $[a, b]$) se reduce a la mitad (y un problema de tamaño $n = 2^k$ se resuelve en k iteraciones). Luego el costo de calcular $\lfloor \sqrt{n} \rfloor$ es $O(\log^3 n)$.
Por lo tanto, conocer $\varphi(n)$ equivale a conocer la factorización de n.

Hallar d sin conocer $\varphi(n)$

Sabemos que existe m entero tal que

$$a^m \equiv a (\bmod n)$$

(en efecto, $m = ed$). Mostraremos un algoritmo probabilístico para factorizar n que es rápido. Luego, si conocemos m podemos factorizar n.
Si $(a, n) = 1$, invertimos a en $\mathbb{Z}_n$ para obtener

$$a^{m-1} \equiv 1 (\bmod n).$$

Necesitamos ahora algunos hechos preliminares:

Lema 5.3.2 Si $a^s \not\equiv 1 (\bmod n)$ para algún $a \in \mathbb{Z}_n^*$, entonces esto ocurre para por lo menos el 50% de los elementos de $\mathbb{Z}_n^*$.

Prueba. Ver la prueba del teorema 2.2.1.

Lema 5.3.3 Si $a^{2s} \equiv 1 (\bmod n)$, para todo $a \in \mathbb{Z}_n^*$ y $b^s \neq \pm 1 (\bmod n)$ para algún $b \in \mathbb{Z}_n^*$, entonces $(b^s - 1, n)$ es p o q.

Prueba. Como $a^{2s} \equiv 1 (\bmod p)$ y $a^{2s} \equiv 1 (\bmod q)$, entonces

$$a^s \equiv \pm 1 (\bmod p) \quad \text{y} \quad a^s \equiv \pm 1 (\bmod q),$$

para todo $a \in \mathbb{Z}_n^*$. Ahora de $b^s \not\equiv \pm 1 (\mathrm{mod}\, n)$, por el teorema chino del resto tenemos

$$b^s \equiv 1 (\mathrm{mod}\, p) \quad \text{y} \quad b^s \equiv -1 (\mathrm{mod}\, q)$$

ó

$$b^s \equiv -1 (\mathrm{mod}\, p) \quad \text{y} \quad b^s \equiv 1 (\mathrm{mod}\, q)$$

(por ejemplo, la posibilidad $b^s \equiv 1 (\mathrm{mod}\, p)$ y $b^s \equiv 1 (\mathrm{mod}\, q)$ implicaría $b^s \equiv 1 (\mathrm{mod}\, n)$, un absurdo). Luego $(b^s - 1, n) \neq 1$, esto es $(b^s - 1, n)$ es p o q.

Lema 5.3.4 Si $a^{2s} \equiv 1 (\mathrm{mod}\, n)$ para todo $a \in \mathbb{Z}_n^*$ y $b^s \not\equiv 1 (\mathrm{mod}\, n)$ para algún $b \in \mathbb{Z}_n^*$, entonces $(a^s - 1, n)$ es p o q para por lo menos el 50% de los elementos de Z_n^*.

Prueba. Observe que $p - 1 \not| s$ o $q - 1 \not| s$ pues de otro modo

$$a^s \equiv 1 (\mathrm{mod}\, p) \quad \text{y} \quad a^s \equiv 1 (\mathrm{mod}\, q)$$

y de ahí $a^s \equiv 1 (\mathrm{mod}\, n)$,para todo $a \in \mathbb{Z}_n^*$, una contradicción.
Supongamos entonces que $p - 1 \not| s$. Sea g una raíz primitiva módulo p, de modo que

$$\mathbb{Z}_p^* = \{g, \ldots, g^{p-1} = 1\}.$$

Como $p - 1 \not| s$ y además $g^{2s} \equiv 1 (\mathrm{mod}\, p)$, entonces

$$g^s \equiv -1 (\mathrm{mod}\, p)$$

(en efecto, lo anterior implica que el orden de g, $p - 1$ divide a s). Por el teorema chino del resto, existe $x \in \mathbb{Z}_n^*$ que satisface

$$x \equiv g (\mathrm{mod}\, p)$$
$$x \equiv 1 (\mathrm{mod}\, q)$$

de donde $x^s \equiv -1 (\mathrm{mod}\, p)$ y $x^s \equiv 1 (\mathrm{mod}\, q)$, i.e.

$$x^s \not\equiv \pm 1 (\mathrm{mod}\, n).$$

Sean $a_1, \ldots, a_t$ los elementos de $\mathbb{Z}_n^*$ para los que se cumple

$$a_i^s \equiv \pm 1 (\mathrm{mod}\, n).$$

Entonces $xa_1, xa_2, \ldots, xa_m$ son distintos y

$$(xa_i)^s = a_i^s x^s \equiv \pm x^s \equiv \pm x^s \not\equiv \pm 1 (\mathrm{mod}\, n).$$

Por lo tanto, por lo menos el %50 de los elementos de $\mathbb{Z}_n^*$ satisfacen

$$a^s \not\equiv \pm 1 (\mathrm{mod}\, n).$$

Algoritmo 5.3.2 Este es un algoritmo de tipo probabilístico para tratar de factorizar n a partir de m.

Entrada: s entero tal que $a^s \equiv 1(\bmod n)$, para $a \in \mathbb{Z}_n^*$
repetir
  $s = s/2$ $[*]$
  $i = 1$
  $f = 1$
  mientras $(i > 0 \wedge i \leq 200)$
    escogemos aleatoriamente a con $1 < a < n$
    si $(a, n) = 1 \wedge a^s \equiv 1(\bmod n)$
      $i = i + 1$
    sino
      $f = (a, n)$
      $i = 0$
mientras $i > 200$ $[**]$
si $f = 1$ $[***]$
  $i = 1$
  mientras $(i > 0 \wedge i \leq 200)$
    escogemos aleatoriamente b con $1 < b < n$
    si $((b, n) = 1 \wedge (b^s - 1, n) = 1)$
      $i = i + 1$
    sino
      $f = \max\{(b, n), (b^s - 1, n)\}$
      $i = 0$
Salida: f es probablemente p o q

Veamos cómo funciona el algoritmo. En el paso $[*]$, s es par pues

$$(-1)^s \equiv 1(\bmod n).$$

Se escogen $a_1, \ldots, a_{200}$ para ver si

$$a_i^s \equiv 1(\bmod n), \quad 1 \leq i \leq 200;$$

si esto es cierto, entonces al llegar a $[**]$, con alta probabilidad tenemos que $a^s \equiv 1(\bmod n)$, para todo $a \in \mathbb{Z}_n^*$ (por el lema 5.3.2, al llegar a $[**]$ existe $a \in \mathbb{Z}_n^*$ con $a^s \not\equiv 1(\bmod n)$ con probabilidad menor que $2^{-200} < 10^{-60}$).
Si en algún momento de la ejecución del primer bloque de código se tiene $(a_i, n) \neq 1$, entonces factorizamos n. Al llegar a $[***]$, si no tenemos un factor de n (si $f = 1$)

tratamos de hallar uno con la ayuda del lema 5.3.4 (en $[***]$ se satisfacen las hipótesis del lema).

Raíz e-ésima

Tratar de calcular $a(\bmod n)$ a partir de $a^e(\bmod n)$, sin conocer $\varphi(n)$ y d. Se piensa que e un problema muy difícil, sin conocer φ ni d. En el presente trabajo no estudiaremos este problema.

Como vemos, existe una relación entre el problema de factorización de números enteros y el problema de romper las claves del método RSA; de hecho, aunque no ha sido aún demostrado, se piensa que ambos problemas son equivalentes. Los algoritmos actuales de factorización no poseen complejidades polinomiales, su velocidad de ejecución está muy por debajo de la velocidad de los algoritmos actuales para determinar primalidad (sus velocidades son exponenciales, o a lo más *subexponenciales*). Esto se traduce en facilidad para generar claves para el método RSA, y dificultad para romperlas. Para más información sobre el problema de factorización, y el *criptoanálisis*, el área de la matemática que estudia las formas de ataque a los métodos de la criptografía, ver [8], [10], [13], [16], [20].

Bibliografía

[1] Diffie, W. y Hellman, M.E. (1976) New Directions in Cryptography, IEEE Trans. Inform. Theory, Vol. IT-22, N° 6, November, 644-654.

[2] Knuth, D.E. (1981) The Art of Computer Programming, Vol. 2, Seminumerical Algorithms, Second Edition, Addison-Wesley Publishing Company, Reading.

[3] Apostol, T.M. (1984) Introducción a la teoría analítica de números, Editorial Reverté, España.

[4] Fouvry, Étienne (1985) Théorème de Brun-Titchmarsh; application au théorème de Fermat. Invent. Math., 79, 383-407.

[5] Jacobson, Nathan (1985) Basic Algebra I, Second Edition, W. H. Freeman and Company, New York.

[6] Koblitz, N. (1987) A Course in Number Theory and Criptography, GMT 114, Springer-Verlag N.Y.

[7] Garcia, A. y Lequain, Y. (1988) Álgebra: um curso de introdução, Instituto de Matemática Pura e Aplicada, IMPA, Rio de Janeiro.

[8] Lemos, M. (1989) Criptografia, Números Primos e Algoritmos, 17° Colóquio Brasileiro de Matemática, IMPA/CNPq

[9] Ribenboim, Paulo (1991) The New Book of Prime Number Records, Springer-Verlag.

[10] Buchmann, J. y Müller, V., Primality testing (1992), Reporte técnico, Universität des Saarlandes.

[11] Buchmann, J. y Müller, V. (1992) Algorithms for factoring integers, Manuscrito, Universität des Saarlandes.

[12] Atkin, A.O.L. y Morain, F. (1993) Eliptic Curves and Primality Testing, Math. Comp., 61, 29-58.

[13] Cohen, Henri (1993) A Course in Computational Algebraic Number Theory, GMT 138, Springer-Verlag Berlin Heidelberg.

[14] Alford, W.R., Granville, A. y Pomerance, C. (1994) There are infinitely many Carmichael numbers, Annals of Mathematics, 140, 743-722.

[15] Baker, R.C. y Harman, C. (1996) The Brun Titchmarsh Theorem on average. In Proceedings of a conference in Honor of Heini Halberstam, Vol. 1, 39-103.

[16] Menezes, A., Van Oorschit P., Vanstone S. (1996) The Handbook of Applied Criptography, CRC Press.

[17] Coutinho, S.C. (1997) Números inteiros e Criptografia RSA, IMPA/SBM

[18] Figuereido, L.M. y Pinto Carvalho, P.C. (2000), Introdução à Geometria Computacional, Monografías del Imca N° 13, Perú.

[19] Chapman, D. (2000) Developing Secure Applications with Visual Basic, Sams Publishing.

[20] Crandall, Richard E. y Pomerance, Carl (2001) Prime Numbers: A Computational Perspective, Springer Verlag N.Y.

[21] Charles Denis Xavier, Primality Testing.

[22] M. Agrawal, N. Kayal y N. Saxena (2002) PRIMES is in P, Reporte de Investigación, Department of Computer Science & Engineering, Indian Institute of Technology Kanpur, India.