Instituto Politécnico Nacional
Escuela Superior de Física y Matemáticas
Departamento de Matemáticas

Congruencias y un Algoritmo de Tiempo Polinomial para la Factorización de Números

Tesis que para obtener el
grado de
Ingeniero Matemático
Presenta:

Christian Michel Curiel Anaya

Director de Tesis: Dr. César Alberto Escobar Gracia

México, D.F. Septiembre de 2008

Resumen

Objetivo: esta tesis tiene como objetivo el mostrar la implementación de un algoritmo de factorización de números primos. Este problema aunque en apariencia sencillo, se vuelve complicado cuando la factorización se da a nivel de números muy grandes.
La factorización de números es importante pues de ella dependen diversas aplicaciones, tales como la implementación de algoritmos para codificar (algoritmos criptográficos), y algunos otros problemas clásicos de la matemática.

El contenido de la tesis se desarrollará de la siguiente manera:

En el primer capítulo se pretende introducir la notación básica, así como los conceptos necesarios para el entendimiento de la parte escencial de este trabajo, se introduce también la notación asintótica, algunos ejemplos en los que se calcula el concepto de complejidad de un algoritmo y algunos conceptos básicos de la teoría de grupos.

El capítulo dos se dedicará al trabajo con congruencias, que aunque es un tema básico nos ayudará al manejo y/ó entendimiento de la aritmética de la que se habla en el capítulo tres. Se enunciaran algunos resultados básicos, en particular se mostrará el teorema chico de Fermat.

En el tercer capítulo se describe el algoritmo de factorización teóricamente más eficiente (salvo algunas mejoras) que actualmente existe, encontrado en Agosto del 2002. Posteriormente se hace una demostración formal de que el proceso realmente es un algoritmo explicando además a que se debe la complejidad que se menciona y las posibles mejoras, para terminar finalmente con la programación en maple del algoritmo original AKS.

Reconocimientos

Agradezco a los Profesores: Adrián Alcántar Torres, Julio César Salas Torres, Luis Alfonso Godinez Contreras, Manuel Robles Bernal, por haber revisado mi tesis y por sus valiosas sugerencias.

A mis amigos con los cuales he pasado momentos de alegría y tristeza, porque se que puedo contar con ellos en todo momento, por su cariño, atención y paciencia muchas gracias, el orden no importará porque todos forman parte especial y primordial en mi vida:

Marcos por convertirse en un hermano no sanguíneo justo en el momento en el que necesitaba tener a una persona tan especial como el a mi lado.

Cesar por no sólo ser un excelente profesor sino también un excelente amigo del cual podré aprender muchas cosas, gracias por apoyarme en éste proyecto y tenerme la confianza y paciencia necesarias para concluirlo satisfactoriamente.

Vicente porque pese al largo tiempo que tenemos de no vernos, en cada reencuentro he recibido palabras de aliento para continuar con mis proyectos.

Ernesto por todas y cada una de las conversaciones que hemos tenido, por su atención y consejos y porque deseo de todo corazón que esos dos luceros que le dan la fortaleza para continuar con su vida pronto los tenga a su lado.

Isaac por los pequeños detalles que hacen de el una gran persona.

Rogelio por el apoyo que recibí de su parte, por la paciencia que me tuvo en todas y cada una de las dudas que se me presentaron a lo largo de mi carrera.

Alejandra por creer en mi, por dejarme conocerla, porque lo que me dice es damasiado valioso y por lo importante e indispensable que se a convertido para mi.

Luis por escucharme, aconsejarme y tranquilizarme en mis momentos de desesperación, por todos y cada uno de los buenos y malos momentos que hemos vivido.

Jorge por siempre estar en todos los momentos porque se que basta una llamada para seber que va a estar escuchando y apoyando con su presencia.

Axinia por contribuir en esta revisión, por ser tan especial y si duda alguna por ser mi favorita, por siempre tener buenos tratos, muchos consejos y una sonrisa.

Xochitl por estar siempre al pendiente de mis avances, y por su preocupación e interés en mi desarrollo personal.

A todos ustedes gracias por ser mis amigos, los mejores amigos que he tenido.

Congruencias y un Algoritmo de Tiempo Polinomial para la Factorización de Números

Septiembre de 2008
Ciudad de México

Christian Michel Curiel Anaya

Dedicatoria

Ésta tesis va dedicada a mi madre María Estela Anaya Velázquez que me dió la vida y que a lo largo del tiempo que llevo de existencia he recibido sus consejos que me han ayudado a ser una mejor persona, a mi padre José Curiel Martínez que día a día me enseña a ser una persona que debe lucha por alcanzar sus metas, a mi hermano Edgar Omar Curiel Anaya que es un ejemplo a seguir y me motiva a continuar estudiando, a mi hermana Ana Laura Curiel Anaya por confiar incondicionalmente en mi y darme palabras de aliento justo en los momentos en los que más lo necesitaba, a mi hermano Oscar Octavio Curiel Anaya que me a apoyado en las decisiones que he tomado y siempre a estado conmigo en los peores momentos para tenderme su mano; aunque ahora te encuentres lejos se que pronto te veré y quiero que sepas que en todo éste tiempo has estado en mi corazón y en mi mente, y a mis sobrinos Ana Ximena Albarran Curiel y Gustavo Adolfo Albarran Curiel porque en cada sonrisa suya me dan felicidad.

Christian Michel Curiel Anaya
Septiembre de 2008
Ciudad de México

Contenido

Capítulo 1

Preliminares

En este capítulo se introducen los conceptos básicos para el entendimiento de este trabajo, así como también las notaciones básicas, y se darán algunos ejemplos que proporcionen un mejor entendimiento.

1.1 Análisis de algoritmos

1.1.1 Notación asintótica

La notación asintótica que se usa para describir un algoritmo de salida de tiempo asintótico se define en términos de funciones cuyo dominio es el conjunto de los números naturales $\mathbb{N} = \{0,\ 1,\ 2, \ldots\}$. Tales notaciones son convenientes para la descripción de un algoritmo de salida de tiempo $T(n)$, usualmente se define sólo con números que de entrada son enteros; y sin duda alguna, a veces es posible excederse en el uso de estas notaciones; por ejemplo, la notación O se puede extender al dominio de los números reales, o, de manera alternativa es posible restringirla a los números naturales.
Esto es importante, sin embargo, para entender el significado exacto, de tal modo que cuando se abuse de la notación ésta no se emplee mal.

1.1.2 Notación Θ

Se establece que en un algoritmo de salida de tiempo, en su peor caso el orden de entrada es $T(n) = \Theta(n^2)$. Se define ahora lo que significa esta notación.

Para una función dada $g(n)$, se denota por $\Theta(g(n))$ el conjunto de funciones

$$\Theta(g(n)) = \{f(n) : \text{existen constantes positivas } c_1,\ c_2 \text{ y } n_0 \text{ tales que}$$

$$0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ para todo } n \geq n_0\}.$$

La función $f(n)$ pertenece al conjunto $\Theta(g(n))$ si existen constantes positivas c_1 y c_2 tal que se encuentran entre $c_1 g(n)$ y $c_2 g(n)$, para un n suficientemente grande.
Aunque $\Theta(g(n))$ es un conjunto, vamos a escribir "$f(n) = \Theta(g(n))$" lo cual indica que $f(n)$ es miembro de $\Theta(g(n))$, o "$f(n) \in \Theta(g(n))$".
De la definición anterior se tiene intuitivamente a las funciones $f(n)$ y $g(n)$, donde $f(n) = \Theta(g(n))$. Para todos los valores de n que están a la derecha de n_0, el valor de $f(n)$ tiende a estar por encima de $c_1 g(n)$ y por debajo de $c_2 g(n)$. En otras palabras para todo $n \geq n_0$, la función $f(n)$ es igual a $g(n)$ dentro de un factor constante. Se dice que $g(n)$ es la frontera asintótica estrecha para $f(n)$.
La definición de $\Theta(g(n))$ requiere que cada miembro de $\Theta(g(n))$ sea asintótico no negativo, esto es, para que $f(n)$ sea no negativo siempre que n sea suficientemente grande. Consecuentemente, la función $g(n)$ en sí misma debe ser asintótica no negativa, o sino el conjunto $\Theta(g(n))$ es vacío. Por lo tanto se asume que cada función usada dentro de la notación Θ es asintótica y no negativa.

1.1.3 Notación *O* mayúscula y *o* minúscula

La notación Θ es una función asintótica con fronteras por encima y por debajo de la función $f(n)$. Cuando sólo se tiene una frontera asintótica superior, se usa la notación O. Para una función $g(n)$ dada, se denota por $O(g(n))$ el conjunto de funciones

$$O(g(n)) = \{f(n) : \text{existen constantes positivas } c \text{ y } n_0 \text{ tales que}$$

$$0 \leq f(n) \leq cg(n) \text{ para todo } n \geq n_0\}.$$

Se usa la notación O para dar una frontera superior en una función, dentro de un factor constante.
Para indicar que la función $f(n)$ es miembro de $O(g(n))$, nuevamente se escribe $f(n) = O(g(n))$. Note que $f(n) = \Theta(g(n))$ implica que $f(n) = O(g(n))$,

puesto que la notación Θ es una notación mas fuerte que la O. Escribiendo teóricamente como conjuntos, se tiene que $\Theta(g(n)) \subseteq O(g(n))$.
Se usa la notación O, cuando se puede describir el tiempo de funcionamiento de un algoritmo, para esto es necesaria la total inspección de la estructura de los algoritmos.
En el caso de la notación o minúscula el límite superior asintótico provisto por la notación O puede o no ser asintóticamente estrecho. Se va a usar esta notación para denotar el límite superior que no es asintóticamente estrecho. Se define formalmente $o(g(n))$ ("o minúscula de g en n") al conjunto de funciones

$$o(g(n)) = \{f(n) : \text{para cualquier constante positiva } c > 0, \text{ existe } n_0 > 0$$

$$\text{tal que } 0 \leq f(n) < cg(n) \text{ para todo } n \geq n_0\}.$$

Las definiciones de la notación O y la notación o son similares. La principal diferencia es que en $f(n) = O(g(n))$, el límite $0 \leq f(n) < cg(n)$ se tiene para alguna constante $c > 0$, pero en $f(n) = o(g(n))$, el límite $0 \leq f(n) < cg(n)$ se tiene para todas las constantes $c > 0$. Intuitivamente, en la notación o, la función $f(n)$ puede hacerse relativamente insignificante a $g(n)$ cuando n tiende a infinito; esto es, $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$.

1.1.4 Notación Ω mayúscula y w minúscula

Justo como la notación O proporciona una frontera asintótica por encima de la función $f(n)$, la notación Ω proporciona una frontera por debajo de la función. Para una función dada $g(n)$, denotaremos $\Omega(g(n))$ al conjunto de funciones

$$\Omega(g(n)) = \{f(n) : \text{existen constantes positivas } c \text{ y } n_0 \text{ tales que}$$

$$0 \leq cg(n) \leq f(n) \text{ para todo } n \geq n_0\}.$$

Análogamente se usa la notación ω minúscula para denotar el límite inferior que no es asintóticamente estrecho. Una forma de definir lo anterior es

$$f(n) \in \omega(g(n)) \text{ si y sólo si } g(n) \in o(f(n)).$$

Se define formalmente $\omega(g(n))$ ("omega minúscula de g en n") al conjunto de funciones

$$\omega(g(n)) = \{f(n) : \text{para cualquier constante positiva } c > 0, \text{ existe } n_0 > 0$$

$$\text{tal que } 0 \leq cg(n) < f(n) \text{ para todo } n \geq n_0\}.$$

La relación $f(n) = \omega(g(n))$ implica que $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$, si es que el límite existe. Esto es, la función $f(n)$ puede hacerse relativamente grande a $g(n)$ cuando n tiende a infinito.

1.1.5 Comparación de funciones

Muchas de las propiedades relacionadas con los números reales se aplican en la comparación de funciones asintóticas. Para ver esta similitud se va a suponer que $f(n)$ y $g(n)$ son asintóticamente positivas.

Transitividad

Si	$f(n) = \Theta(g(n))$	y	$g(n) = \Theta(h(n))$	implica	$f(n) = \Theta(h(n))$
Si	$f(n) = O(g(n))$	y	$g(n) = O(h(n))$	implica	$f(n) = O(h(n))$
Si	$f(n) = \Omega(g(n))$	y	$g(n) = \Omega(h(n))$	implica	$f(n) = \Omega(h(n))$
Si	$f(n) = o(g(n))$	y	$g(n) = o(h(n))$	implica	$f(n) = o(h(n))$
Si	$f(n) = \omega(g(n))$	y	$g(n) = \omega(h(n))$	implica	$f(n) = \omega(h(n))$.

Reflexividad

$$f(n) = \Theta(f(n))$$
$$f(n) = O(f(n))$$
$$f(n) = \Omega(f(n)).$$

Simetría

$$f(n) = \Theta(g(n)) \text{ si y sólo si } g(n) = \Theta(f(n)).$$

Simetría transpuesta

$$f(n) = O(g(n)) \text{ si y sólo si } g(n) = \Omega(f(n))$$
$$f(n) = o(g(n)) \text{ si y sólo si } g(n) = \omega(f(n)).$$

De esta manera se puede ver la analogía ($\approx$) que hay entre la comparación asintótica de dos funciones, f y g y la comparación de dos números reales, a y b.

$$\begin{array}{rcl} f(n) = O(g(n)) & \approx & a \leq b \\ f(n) = \Omega(g(n)) & \approx & a \geq b \\ f(n) = \Theta(g(n)) & \approx & a = b \\ f(n) = o(g(n)) & \approx & a < b \\ f(n) = \omega(g(n)) & \approx & a > b. \end{array}$$

Sin embargo la propiedad de tricotomía no se cumple para la notación asintótica.

1.1.6 Notaciones estandar y funciones comunes

Funciones monótonas: la función $f(n)$ es monótonamente creciente si $m \leq n$ implica $f(m) \leq f(n)$. Análogamente, es monótonamente decreciente si $m \leq n$ implica $f(m) \geq f(n)$. La función $f(n)$ es estrictamente creciente si $m < n$ implica $f(m) < f(n)$ y es estrictamente decreciente si $m < n$ implica $f(m) > f(n)$.

Funciones polinomiales: dado un entero positivo d, el polinomio n de grado d es una función $p(n)$ de la forma

$$p(n) = \sum_{i=0}^{d} a_i n^i$$

donde las constantes $a_0,\ a_1, \ldots,\ a_d$ son los coeficientes del polinomio y $a_d \neq 0$. El polinomio es asintóticamente positivo si y sólo si $a_d > 0$. Para un polinomio asintóticamente positivo $p(n)$ de grado d, se tiene que $p(n) = \Theta(n^d)$. Para cualquier constante real $a \geq 0$, la función n^a es monótonamente creciente, y para cualquier constante real $a \leq 0$, la función n^a es monótonamente decreciente. Por lo que se puede decir que la función $f(n)$ es polinomialmente acotada si $f(n) = n^{O(1)}$, lo cual es equivalente a decir que $f(n) = O(n^k)$ para algunas constantes k.

Funciones exponenciales: para todo número real $a \neq 0$, m y n, se tienen las siguientes identidades

$$\begin{aligned} a^0 &= 1 \\ a^1 &= a \\ a^{-1} &= \frac{1}{a} \\ (a^m)^n &= a^{mn} \\ (a^m)^n &= (a^n)^m \\ a^m a^n &= a^{m+n}. \end{aligned}$$

Para todo n y $a \geq 1$, la función a^n es monótonamente creciente en n. El valor de crecimiento de las polinomiales y exponenciales está relacionado por

el hecho de que para todas las constantes reales a y b tal que $a > 1$, se tiene que

$$\lim_{n\to\infty} \frac{n^b}{a^n} = 0 \tag{1.1}$$

por lo cual se concluye que $n^b = o(a^n)$.
Así que, cualquier función exponencial positiva crece mas rápidamente que cualquier función polinomial.
Se usará e para denotar $2.71828\ldots$, la función base del logaritmo natural, por otra parte se tiene que para todo número real x

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots = \sum_{i=0}^{\infty} \frac{x^i}{i!}$$

así como también se cumple la siguiente desigualdad

$$e^x \geq 1 + x$$

donde la igualdad se cumple si y sólo si $x = 0$. Cuando $|x| \leq 1$, se tiene la siguiente aproximación

$$1 + x \leq e^x \leq 1 + x + x^2$$

cuando $x \to 0$, la aproximación de e^x por $1 + x$ es verdaderamente buena ya que $e^x = 1 + x + \Theta(x^2)$, por lo que se tiene entonces que para toda x, $\lim_{n\to\infty} \left(1 + \frac{x}{n}\right)^n = e^x$.

Funciones logarítmicas: en este caso se usará la siguiente notación

$$\begin{aligned} \lg n &= \log_2 n \\ \ln n &= \log_e n \\ \lg^k n &= (\lg n)^k \\ \lg \lg n &= \lg(\lg n). \end{aligned}$$

Para $n > 0$ y $b > 1$, la función $\log_b n$ es estrictamente creciente. Para todo

número real $a > 0$, $b > 0$, $c > 0$ y n

$$\begin{aligned} a &= b^{\log_b a} \\ \log_c(ab) &= \log_c a + \log_c b \\ \log_b a^n &= n \log_b a \\ \log_b a &= \frac{\log_c a}{\log_c b} \\ \log_b \left(\frac{1}{a}\right) &= -\log_b a \\ \log_b a &= \frac{1}{\log_a b} \\ a^{\log_b n} &= n^{\log_b a}. \end{aligned}$$

Ya que cambiando la base del logaritmo por alguna otra constante solamente cambiará el valor del logaritmo en un factor constante, a menudo se usará la notación "$\lg n$" para no darle importancia a los factores constantes, como en la notación O.
Una expansión simple en serie de potencias para $\ln(1 + x)$ cuando $|x| < 1$ es:

$$\ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \frac{x^5}{5} - \dots$$

en donde también se cumple la siguiente desigualdad para $x > -1$

$$\frac{x}{1+x} \leq \ln(1+x) \leq x$$

obteniéndose la igualdad si y sólo si $x = 0$.
La función $f(n)$ se llamará límite polilogarítmico si $f(n) = \lg^{O(1)} n$. Se puede relacionar el crecimiento de una función polinomial y una polilogarítmica si sustituimos $\lg n$ por n y 2^a por a en la ecuación (1.1), obteniendo así que el $\lim_{n\to\infty} \frac{\lg^b n}{2^{a \lg n}} = \lim_{n\to\infty} \frac{\lg^b n}{n^a} = 0$ de este limite se concluye que $\lg^b n = o(n^a)$ para cualquier constante $a > 0$. Así que, cualquier función polinomial positiva crece más rápidamente que cualquier función polilogarítmica.

Ejemplo 1.1.1 Estimando el tiempo utilizado para calcular la inversa de una

matriz A si es que existe, $A \in M_{m\times m}(\mathbb{R})$ por el método de Gauss Jordan

$$\left[\begin{array}{cccccc|cccccc} a_{11} & a_{12} & \cdots & a_{1k} & \cdots & a_{1n} & 1 & 0 & \cdots & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & a_{2k} & \cdots & a_{2n} & 0 & 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{k1} & a_{k2} & \cdots & a_{kk} & \cdots & a_{kn} & 0 & 0 & \cdots & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nk} & \cdots & a_{nn} & 0 & 0 & \cdots & 0 & \cdots & 1 \end{array}\right]$$

en el paso k-ésimo de la eliminación primero se divide cada entrada a_{kj}, $j \geq k$ del renglón k-ésimo por a_{kk} lo cual llevará $2n-k$ operaciones para así obtener una matriz de la forma

$$\left[\begin{array}{cccccc|cccccc} a_{11} & a_{12} & \cdots & a_{1k} & \cdots & a_{1n} & 1 & 0 & \cdots & 0 & \cdots & 0 \\ a_{21} & a_{22} & \cdots & a_{2k} & \cdots & a_{2n} & 0 & 1 & \cdots & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \dfrac{a_{k1}}{a_{kk}} & \dfrac{a_{k2}}{a_{kk}} & \cdots & 1 & \cdots & \dfrac{a_{kn}}{a_{kk}} & \dfrac{0}{a_{kk}} & \dfrac{0}{a_{kk}} & \cdots & \dfrac{1}{a_{kk}} & \cdots & \dfrac{0}{a_{kk}} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nk} & \cdots & a_{nn} & 0 & 0 & \cdots & 0 & \cdots & 1 \end{array}\right]$$

para cada renglón $1 \leq j \leq n$, $j \neq k$ se le suma $-a_{kk}$ veces el renglón k-ésimo lo cual lleva $2n-k$ multiplicaciones y $2n-k$ sumas, así que en el paso completo se realizan $3(2n-k)$ operaciones.
Ahora como $1 \leq k \leq n$ en total se realizan

$$\underbrace{3(2n+1)+3(2n+2)+\cdots+3(2n+n)}_{n\ veces}$$

$$\begin{aligned} &= 3((2n-1)+(2n-2)+\cdots+(2n-n)) \\ &= 3((2n)n-(1+2+\cdots+n)) \\ &= 3\left(2n^2-\frac{n(n+1)}{2}\right) \\ &= 3\left(2n^2-\frac{n^2}{2}-\frac{n}{2}\right) \\ &= O(n^2) \end{aligned}$$

es decir, el algoritmo tiene orden $O(n^2)$.

Ejemplo 1.1.2 Ahora suponiendo que se quiere calcular la n-ésima potencia de un entero x (x^n).
Se puede intentar calcular x^{n-1} y multiplicar por x, siguiendo este algoritmo se tienen $n-1$ operaciones, es decir, el algoritmo descrito es de orden $O(n)$.
En este caso se describirá otro algoritmo llamado algoritmo binario para el cálculo de potencias.
Suponga que n está expresado en forma binaria, y en ésta expresión binaria se cambiarán los 1's por sx's y los 0's por s's, posteriormente se borra la primera sx e interpretamos la x como multiplicar por x y la s como elevar al cuadrado. Por ejemplo tomando $n = 15$, se tiene que su expresión binaria es 1111, y realizando los cambios antes mencionados se tiene

$$sxsxsxsx$$

$$sxsxsx$$

$$\underbrace{x \to x^2 \to x^3 \to x^6 \to x^7 \to x^{14} \to x^{15}}_{6\ operaciones}$$

en este caso lo que se hizo fue calcular la representación binaria del exponente, lo que quiere decir que para el caso en el que $n = 15$ se tiene que $15 = 1 + 2 + 2^2 + 2^3$ ($15 = (1111)_2$), y de una manera más general si se quiere calcular x^n y $n = n_0 + 2n_1 + 2^2 n_n + \cdots 2^{k-1} n_{k-1} + 2^k$ se tendrá que

$$x^n = x^{n_0} \cdot x^{2n_1} \cdot x^{2^2 n_2} \cdots x^{2^{k-1} n_{k-1}} \cdot x^{2^k}$$

en donde se realizan k multiplicaciones para calcular $x^2,\ x^4, \ldots,\ x^{2^k}$ y después, tantas como unos tenga la expresión binaria del exponente menos una, por lo que la complejidad de este algoritmo es $\log_2 n$, es decir, es de orden $O(\log n)$.

1.1.7 Complejidad de problemas

Una clase de complejidad es un conjunto de problemas para los que se observa una determinada propiedad relacionada con su complejidad. El objetivo de esta sección es agrupar los problemas de decisión en clases atendiendo a su dificultad, es decir, a la complejidad computacional del mejor algoritmo capaz de solucionarlos.

La complejidad de un problema se clasifica en términos del costo de solucionar las instancias más difíciles con el mejor algoritmo posible (usando una máquina

de Turing o equivalente).

Alan Turing definió un modelo abstracto de computador que permite dotar de un significado preciso al concepto de algoritmo: la máquina de Turing, un dispositivo formal consistente en una cinta formada por celdas (cada una de las cuales puede albergar un símbolo), un cabezal de lectura-escritura capaz de desplazarse a izquierda y derecha sobre la cinta, un conjunto finito de estados en los que se puede encontrar la máquina (uno de los cuales es el estado inicial) y una tabla de acciones que dice que símbolo escribir, que desplazamiento debe efectuar el cabezal, y en que estado debe quedar la máquina en función del símbolo leído y del estado actual. Cada máquina de Turing implementa una determinada función parcial de las cadenas de entrada (configuraciones iniciales de la cinta) en las cadenas de salida.

La idea es que todo programa escrito en un lenguaje de programación puede traducirse a una máquina de Turing y toda máquina de Turing puede codificarse en un lenguaje de programación suficientemente potente.

A la hora de comparar diferentes algoritmos, interesa estudiar su eficiencia computacional, esto es, el mayor o menor aprovechamiento de los recursos disponibles. Los dos recursos habitualmente estudiados son el tiempo y el espacio (memoria).

La primera familia de interés que se encarga del estudio de la complejidad en los problemas es la familia o clase P: el conjunto de problemas de decisión resolubles en tiempo polinómico por una máquina de Turing determinista (esto es, por un computador convencional). La letra que los identifica es la inicial de polinómico. La clase P está cerrada bajo complemento, es decir, los problemas de decisión complementarios de problemas de decisión en P son también problemas de P. Otra familia de problemas de interés es la clase NP: problemas para los que existe una máquina de Turing no determinista capaz de dar respuesta en tiempo polinómico, pero para los que no se conoce máquina de Turing determinista que haga lo mismo en tiempo polinómico. Hay una relación evidente entre las clases P y NP: todo problema P es un problema NP, es decir, $P \subseteq NP$: podemos usar el propio algoritmo P como generador y comprobador de certificados, pues funciona en tiempo polinómico sobre una máquina determinista.

Si $f(n)$ es un polinomio $a_t n^t + a_{t-1} n^{t-1} + \cdots + a_1 n + a_0$, entonces $f(n) = O(n^t)$. Si $T = O(n^t)$ para alguna constante t, se dice que el algoritmo es de complejidad polinomial. Si $t = 0$, es constante; si $t = 1$, es lineal; si $t = 2$, es cuadrático, etc. Si $T = O(t^{h(n)})$, para t constante y $h(n)$ polinomio, el algoritmo es exponencial; este tipo de problemas suelen requerir enormes cantidades de pasos, por lo que los algoritmos de tiempo exponencial se consideran, en general, de escaso o nulo interés práctico.

Si el algoritmo es polinomial, el problema se define como tratable. A la clase de problemas tratables se les llama P. A los no tratables también se les llaman difíciles o duros. A la clase de problemas duros que se les llama NP.

Los problemas NP son de gran interés, pues no se conoce ningún algoritmo de tiempo polinómico para ellos, pero tampoco se ha demostrado que tal algoritmo no exista; estos problemas presentan una interesante propiedad: si se proporciona un certificado de solución para cada instancia del problema, son capaces de resolver el problema en tiempo polinómico. La dificultad estriba en cómo generar eficientemente esos certificados, algo para lo que no se conoce algoritmo de tiempo polinómico alguno.

Es evidente, como se vio, en el ejemplo 1.1.1 el cálculo de la inversa de una matriz cuadrada requiere, al menos, un tiempo $O(n^2)$, ya que hay que especificar el valor de n^2 celdas. Esto no significa que se pueda construir un algoritmo $O(n^2)$ para resolver el problema; el algoritmo asintóticamente más eficiente que se conoce requiere tiempo $O(n^{2.376})$. Éste es un problema que puede resolverse en tiempo polinómico y, por lo tanto, se le puede dar solución mediante el uso de computadoras. Otros problemas sólo pueden resolverse en tiempo exponencial, pues expresar la propia solución consume una cantidad de espacio exponencial (por ejemplo, enumerar las $n!$ permutaciones de n elementos). De otros problemas se sabe que no es posible diseñar un algoritmo que proporcione una solución para toda instancia del problema (por ejemplo, decidir si cualquier programa se para ante toda posible entrada de datos). Estos últimos dos ejemplos son problemas difíciles.

1.2 La transformada discreta de Fourier

La serie de Fourier es álgebra lineal en dimensiones infinitas. Los "vectores" son funciones $f(x)$; éstas son proyectadas sobre los senos y los cosenos. Así se obtienen los coeficientes de Fourier a_k y b_k. A partir de esta serie infinita de senos y cosenos, multiplicados por a_k y b_k, es posible reconstruir a $f(x)$. Éste es el caso clásico, en el que soñaba Fourier, aunque en los cálculos verdaderos lo que se calcula es la transformada discreta de Fourier.

Estas operaciones están basadas en la ortogonalidad. La entrada es una sucesión de números $y_0, \ldots, y_{n-1}$, en vez de una función $f(x)$. La salida $c_0, \ldots, c_{n-1}$ tiene la misma longitud n. La relación entre y y c es lineal, de modo que debe estar dada por una matriz. Esta es la matriz F de Fourier.

Definición 1.2.1 *La transformada rápida de Fourier denotada por F es la transformación lineal $F : \mathbb{C}^n \to \mathbb{C}^n$ cuya matriz asociada es*

$$F = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & w & w^2 & \cdots & w^{n-1} \\ 1 & w^2 & w^4 & \cdots & w^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & w^{n-1} & w^{2(n-1)} & \cdots & w^{(n-1)^2} \end{bmatrix}$$

en donde w es una raíz n-ésima de la ecuación $w^n = 1$, $w \neq 1$. Mas aún $w = e^{\frac{2\pi}{n}i} = \cos\dfrac{2\pi}{n} + i\text{sen}\dfrac{2\pi}{n}$.

En la siguiente sección se tratará de ver más a fondo el cálculo de la raíz n-ésima de la ecuación $w^n = 1$.

1.2.1 Raíces complejas de la unidad

Todo polinomio real o complejo de grado n tiene un conjunto completo de n raíces (que pueden ser complejas, reales o reales repetidas). Este es el llamado teorema fundamental del álgebra.

En esta sección se tiene cierto interés en ecuaciones como $x^4 = 1$ ya que son las que nos llevan al cálculo de la transformada rápida de Fourier. Ésta tiene cuatro soluciones: las raíces cuartas de la unidad. Las dos raíces cuadradas

de la unidad son 1 y -1. Las raíces cuartas son las raíces cuadradas de las raíces cuadradas, 1 y -1, i y $-i$. El número i satisface $i^4 = 1$. Para calcular las raíces octavas de la unidad se requieren las raíces cuadradas de i, lo cual lleva a $w = \dfrac{(1+i)}{\sqrt{2}}$. Al elevar al cuadrado a w se obtiene $\dfrac{1+2i+i^2}{2}$, que es i porque $1 + i^2$ es cero. Así, $w^8 = i^4 = 1$.

Los números complejos $\cos\theta + i\text{sen}\theta$ en la matriz de Fourier son extremadamente especiales y de suma importancia para los cálculos mencionados posteriormente. La parte real se traza sobre el eje x y la parte imaginaria, sobre el eje y. Así, el número w está sobre la circunferencia unitaria; su distancia al origen es $\cos^2\theta + \text{sen}^2\theta = 1$; el cual forma un ángulo θ con la horizontal.

El cuadrado de w puede encontarse directamente (simplemente duplicando el ángulo):

$$w^2 = (\cos\theta + i\text{sen}\theta)^2 = \cos^2\theta - \text{sen}^2\theta + 2i\text{sen}\theta\cos\theta \tag{1.2}$$

La parte real de $\cos^2\theta - \text{sen}^2\theta$ es $\cos 2\theta$, y la parte imaginaria $2\text{sen}\theta\cos\theta$ es $\text{sen}2\theta$. Por tanto, $w^2 = \cos 2\theta + i\text{sen}2\theta$. El cuadrado de w sigue estando en la circunferencia unitaria, pero ahora a un ángulo de 2θ. De manera análoga w^n cumple $w^n = \cos n\theta + i\text{sen}n\theta$.

Hay una mejor manera de tomar potencias de w. La combinación del coseno y el seno es una exponencial compleja, con amplitud 1 y ángulo de fase θ:

$$\cos\theta + i\text{sen}\theta = e^{i\theta} \tag{1.3}$$

Las reglas para multiplicar, como por ejemplo $(e^2)(e^3) = e^5$, se siguen cumpliendo cuando los exponentes $i\theta$ son imaginarios.

$$w^2 = e^{i2\theta},\ w^n = e^{in\theta},\ \frac{1}{w} = e^{-i\theta} \tag{1.4}$$

Con esta fórmula es posible resolver $w^n = 1$. Esto se convierte en $e^{in\theta} = 1$, de modo que $n\theta$ debe llevarnos alrededor de la circunferencia unitaria y volver al principio. Por lo cual la solución a escoger es $\theta = 2\pi n$: la n-ésima raíz "primitiva" de la unidad es:

$$w_n = e^{\frac{2\pi i}{n}} = \cos\frac{2\pi}{n} + i\text{sen}\frac{2\pi}{n} \tag{1.5}$$

como ya se había mencionado en la sección anterior.
La matriz F juega un papel muy importante en el ahorro del tiempo y de operaciones para la elaboración del programa que se tratará de explicar en los capítulos posteriores, ya que toda la tecnología del procesamiento de señales digitales depende de ella.

1.2.2 La matriz de Fourier y su inversa

Una relación importante que hay entre la matriz inversa de F y F conjugada $(\overline{F})$ es la que se menciona en el siguiente teorema.

Teorema 1.2.2 $F^{-1} = \dfrac{\overline{F}}{n}$ *donde* $\overline{F}$ *es la matriz conjugada de* F.

Demostración Realizando el producto $F\overline{F}$ se tiene

$$\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & w & w^2 & \cdots & w^{n-1} \\ 1 & w^2 & w^4 & \cdots & w^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & w^{n-1} & w^{2(n-1)} & \cdots & w^{(n-1)^2} \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \overline{w} & \overline{w}^2 & \cdots & \overline{w}^{n-1} \\ 1 & \overline{w}^2 & \overline{w}^4 & \cdots & \overline{w}^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \overline{w}^{n-1} & \overline{w}^{2(n-1)} & \cdots & \overline{w}^{(n-1)^2} \end{bmatrix} =$$

$$\begin{bmatrix} n & \cdots & 1+\overline{w}^{n-1}+\overline{w}^{2(n-1)}+\cdots+\overline{w}^{(n-1)^2} \\ 1+w+w^2+\cdots+w^{n-1} & \cdots & 1+\overline{w}^{n}+\overline{w}^{2n}+\cdots+\overline{w}^{n(n-1)} \\ \vdots & \ddots & \vdots \\ 1+w^{n-1}+w^{2(n-1)}+\cdots+w^{(n-1)^2} & \cdots & n \end{bmatrix}$$

Ahora $(1+w+w^2+\cdots+w^{n-1})(1-w) = 1-w^n$ de donde $w^n = 1$ por lo tanto $1+w+w^2+\cdots+w^{n-1} = 0$, $(1+w^{n-1}+w^{2(n-1)}+\cdots+w^{(n-1)^2})(1-w^{n-1}) = 1-(w^{n-1})^n = 1-\left(\left(e^{\frac{2\pi}{n}i}\right)^{n-1}\right)^n = 1-e^{(2\pi i)(n-1)} = 1-1^{n-1} = 0$ por lo tanto $1+w^{n-1}+w^{2(n-1)}+\cdots+w^{(n-1)^2} = 0$, análogamente para cada una de las entradas se tiene que

$$F\overline{F} = \begin{bmatrix} n & 0 & 0 & \cdots & 0 \\ 0 & n & 0 & \cdots & 0 \\ 0 & 0 & n & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & n \end{bmatrix}$$

luego $F\left(\frac{\overline{F}}{n}\right) = I$ y por lo tanto $F^{-1} = \frac{\overline{F}}{n}$. □

Por esta razón F^{-1} es invertible y calcularla resulta muy fácil así como también las multiplicaciones por F y F^{-1} deben ser rápidas.

Estas afirmaciones son validas; ya que F es simétrica y ortogonal (excepto por un factor $\sqrt{n}$), y sólo tiene un inconveniente: sus elementos son números complejos. Este inconveniente se minimiza por el hecho de que todos los elementos de F y F^{-1} son potencias de un solo número w, el cual se mencionó anteriormente.

Ejemplo 1.2.3 Así que la matriz

$$F = \begin{bmatrix} 1 & 1 & 1 \\ 1 & e^{\frac{2\pi i}{3}} & e^{\frac{4\pi i}{3}} \\ 1 & e^{\frac{4\pi i}{3}} & e^{\frac{8\pi i}{3}} \end{bmatrix}$$

tiene como matriz inversa a

$$F^{-1} = \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 1 & e^{-\frac{2\pi i}{3}} & e^{-\frac{4\pi i}{3}} \\ 1 & e^{-\frac{4\pi i}{3}} & e^{-\frac{8\pi i}{3}} \end{bmatrix}.$$

En el caso continuo, la serie de Fourier puede reproducir a $f(x)$ sobre todo un intervalo utilizando una infinidad de senos y cosenos (o exponenciales). Esto es

$$f(x) = c_0 + c_1 e^{ix} + \cdots + c_n e^{nix}$$

o de manera análoga

$$f(x) = c_0 + \sum a_n \cos nx + \sum b_n i\text{sen}nx.$$

En el caso discreto, basta tomar $n+1$ valores de f en $n+1$ puntos digamos $f(x_i)$, $1 \leq i \leq n$ con x_i enteros.

Ejemplo 1.2.4 Sea

$$f(x) = c_0 + c_1(\cos x + i\text{sen}x) + c_2(\cos 2x + i\text{sen}2x) + c_3(\cos 3x + i\text{sin}3x)$$

se quieren encontrar $c_0,\ c_1,\ c_2,\ c_3$ tal que $f(0) = 2,\ f\left(\frac{\pi}{2}\right) = 4,\ f(\pi) = 6,\ f\left(\frac{3\pi}{2}\right) = 8$ así que el problema completo es calcular $Fc = y$ y este se resuelve si y sólo si

$$\begin{array}{ccccccccc} c_0 & + & c_1 & + & c_2 & + & c_3 & = & 2 \\ c_0 & + & ic_1 & + & i^2c_2 & + & i^3c_3 & = & 4 \\ c_0 & + & i^2c_1 & + & i^4c_2 & + & i^6c_3 & = & 6 \\ c_0 & + & i^3c_1 & + & i^6c_2 & + & i^9c_3 & = & 8 \end{array}$$

o de manera análoga

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & i^2 & i^3 \\ 1 & i^2 & i^4 & i^6 \\ 1 & i^3 & i^6 & i^9 \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \\ 6 \\ 8 \end{bmatrix}$$

es decir

$$F \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \\ 6 \\ 8 \end{bmatrix}$$

esto implica que

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = F^{-1} \begin{bmatrix} 2 \\ 4 \\ 6 \\ 8 \end{bmatrix} = \frac{\overline{F}}{4} \begin{bmatrix} 2 \\ 4 \\ 6 \\ 8 \end{bmatrix}$$

lo que significa que

$$\frac{1}{4} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -i & -1 & i \\ 1 & -1 & 1 & -1 \\ 1 & i & -1 & -i \end{bmatrix} \begin{bmatrix} 2 \\ 4 \\ 6 \\ 8 \end{bmatrix} = \begin{bmatrix} 5 \\ -1+i \\ -1 \\ -1-i \end{bmatrix}$$

por lo tanto

$$f(x) = 5+(-1+i)(\cos x + i\text{sen}x) - (\cos 2x + i\text{sen}2x) + (-1-i)(\cos 3x + i\text{sen}3x)$$

o lo que es lo mismo

$$f(x) = 5 + (-1+i)e^{ix} - e^{2ix} + (-1-i)e^{3ix}$$

de donde se comprueba que

$$f(0) = 5 + (-1+i) - 1 + (-1-i) = 2$$
$$f\left(\frac{\pi}{2}\right) = 5 + (-1+i)e^{i\frac{\pi}{2}} - e^{i\pi} + (-1-i)e^{3i\frac{\pi}{2}} = 4$$
$$f(\pi) = 5 + (-1+i)e^{i\pi} - e^{2i\pi} + (-1-i)e^{3i\pi} = 6$$
$$f\left(\frac{3\pi}{2}\right) = 5 + (-1+i)e^{i\frac{3\pi}{2}} - e^{3i\pi} + (-1-i)e^{9i\frac{\pi}{2}} = 8.$$

Para toda n, la matriz que relaciona y con c puede invertirse. Para encontrar las c's es necesario invertir F. En este caso la matriz fue de 4 por 4, F^{-1} se construyó a partir de $\frac{1}{i} = -i$. Ésta es la regla general, que F^{-1} proviene del número complejo $w^{-1} = \overline{w}$. El cual está en el ángulo $-\dfrac{2\pi}{n}$, donde w estaba en el ángulo $+\dfrac{2\pi}{n}$.

Las multiplicaciones que se realizaron en el ejemplo anterior de F y F^{-1} pueden hacerse de manera extraordinariamente rápida e ingeniosa. En vez de realizar n^2 multiplicaciones por separado, que provienen de los n^2 elementos de la matriz, para efectuar los productos matrices-vectores Fc y $F^{-1}y$, sólo se requieren $\frac{1}{2}n\log n$ pasos. Este reordenamiento de la multiplicación se denomina transformada rápida de Fourier.

1.2.3 La transformada rápida de Fourier

Como vimos anteriormente Fc y F^{-1} pueden calcularse rápidamente. La clave está en la relación de F_4 y F_2, o mejor aún, con dos copias de F_2, que van a la matriz F_2^*:

$$F_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & i^2 & i^3 \\ 1 & i^2 & i^4 & i^6 \\ 1 & i^3 & i^6 & i^9 \end{bmatrix}$$

está próxima a

$$F_2^* = \begin{bmatrix} 1 & 1 & * & * \\ 1 & -1 & * & * \\ * & * & 1 & 1 \\ * & * & 1 & -1 \end{bmatrix}$$

F_4 contiene las potencias de $w_4 = i$, la raíz cuarta de 1. F_2^* contiene las potencias de $w_2 = -1$, la raíz cuadrada de 1. La transformada de 2 por 2, aplicada dos veces, sólo requiere la mitad de trabajo que una transformada directa de 4 por 4. Si una transformada de 64 por 64 puede sustituirse por dos transformadas de 32 por 32, entonces el trabajo se reduce a la mitad. Lo que hace realidad, y posible en la práctica y servirá para reducir los cálculos en el futuro desarrollo del algoritmo, lo anterior es la simple relación entre w_{64} y w_{32}:

$$(w_{64})^2 = w_{32} \text{ o bien } \left(e^{\frac{2\pi i}{64}}\right)^2 = e^{\frac{2\pi i}{32}}.$$

La trigésimosegunda raíz está dos veces más lejos en la circunferencia unitaria que la sexagésimacuarta raíz. Si $w^{64} = 1$, entonces $(w^2)^{32} = 1$. La m-ésima raíz es el cuadrado de la n-ésima raíz, si m es la mitad de n:

$$w_n^2 = w_m \text{ si } m = \frac{1}{2}n.$$

La rapidez de la TRF, depende de trabajar con números altamente compuestos como $2^{10} = 1024$. Sin la transformada rápida, se requieren $(1024)^2$ multiplicaciones para obtener F por c. En contraste, una transformada rápida puede realizar cada multiplicación en sólo $(5)(1024)$ pasos. Es 200 veces más rápida, ya que sustituye un factor de 1024 por 5. En general sustituye n^2 multiplicaciones por $\frac{1}{2}nl$, cuando n es 2^l. Al relacionar F_n con dos copias de $F_{\frac{n}{2}}$, y luego con cuatro copias de $F_{\frac{n}{4}}$, y finalmente con una F muy pequeña, los n^2 pasos de costumbre se reducen a $\frac{1}{2}n \log n$.

Es necesario ver como $y = F_n c$ (un vector de n componentes) puede calcularse a partir de dos vectores que sólo miden la mitad de largo. El primer paso es dividir c entre si mismo, separando sus componentes con número par de sus componentes con número impar:

$$c' = c_0, c_2, \ldots, c_{n-2} \quad y \quad c'' = c_1, c_3, \ldots, c_{n-1}.$$

Los coeficientes simplemente van de forma alterna en c' y c''. A partir de estos vectores, la transformada a la mitad de tamaño proporciona $y' = F_m c'$ y $y'' = F_m c''$. Estas son las dos multiplicaciones por la matriz más pequeña F_m. El problema central consiste en calcular y a partir de los vectores y' y y'' que miden la mitad, Cooley y Tukey se dieron cuenta de como hacerlo:

Las m primeras y las m últimas componentes del vector $y = F_n c$ son

$$\begin{aligned} y_j &= y'_j + w_n^j y''_j \quad , \quad j = 0, \ldots, m-1 \\ y_{j+m} &= y'_j - w_n^j y''_j \quad , \quad j = 0, \ldots, m-1. \end{aligned}$$

Por tanto, los tres pasos son: separar c en c' y c''; transformarlos por medio de F_m en y' y y''; y reconstruir y a partir de la ecuación anterior.

Para comprobar las ecuaciones anteriores (comenzando con la primera), es necesario separar y_j en par e impar:

$$y_j = \sum_{k=0}^{n-1} w_n^{jk} c_k \text{ es idéntica a } \sum_{k=0}^{m-1} w_n^{2kj} c_{2k} + \sum_{k=0}^{m-1} w_n^{(2k+1)j} c_{2k+1}.$$

Cada sumatoria de la derecha consta de $m = \frac{1}{2}n$ términos. Debido a que w_n^2 es w_m, las dos sumatorias son:

$$y_j = \sum_{k=0}^{m-1} w_m^{kj} c'_k + w_n^j \sum_{k=0}^{m-1} w_m^{kj} c''_k = y'_j + w_n^j y''_j.$$

Para la segunda ecuación, al tenerse el subíndice $j+m$ en vez del subíndice j produce un cambio de signo:

Dentro de las sumatorias, para $w_m^{k(j+m)}$ permanecerá w_m^{kj} debido a que $w_m^{km} = 1^k = 1$. Entonces $w_n^{j+m} = -w_n^j$ ya que $w_n^m = e^{\frac{2\pi i m}{n}} = e^{\pi i} = -1$. De este modo se resolvio el problema de calcular $y = F_n c$.

1.3 Generalidades de grupos

Es bien sabido que al considerar dos enteros a y b, el cociente de a por b no siempre deja residuo cero, lo que da lugar al concepto de divisibilidad, uno de los más importantes en la teoría de números. Por lo que se tienen las siguientes definiciones y teoremas.

Definición 1.3.1 *Sea $p \in \mathbb{Z}$ entonces p es primo si cuando $p \mid ab$ se tiene que $p \mid a$ ó $p \mid b$.*

Definición 1.3.2 *Sean $a,\ b \in \mathbb{Z}^+$ el máximo común divisor de a y b denotado por (a,b) es d si:*

1. *$d \mid a$ y $d \mid b$.*
2. *Si $e \mid a$ y $e \mid b$ entonces $e \mid d$, $(e > 0)$.*

Teorema 1.3.3 *Si $d = (a,b)$ entonces $d = ax + by$ para algunos enteros x, y.*

Teorema 1.3.4 *[Algoritmo de la división] Dados $a,\ b \in \mathbb{Z}$, $a \neq 0$ existen $q,\ r \in \mathbb{Z}$ tal que $b = aq + r$ con $r = 0$ ó $0 < r < |a|$*

Teorema 1.3.5 *[Algoritmo de Euclides] Sean $a,\ b \in \mathbb{Z}$, $a,\ b \neq 0$ el siguiente proceso calcula (a,b) (el mcd):*

1. *Suponiendo $a,\ b > 0$.*
2. *Por el algoritmo de la división*

$$b = aq_1 + r_1$$

   si $r_1 = 0$ se acaba, en este caso $(a,b) = a$, si $r_1 \neq 0$ siga.

3. *tome $b = a$ y $a = r_1$ y aplique nuevamente el paso uno*

$$\begin{array}{cc} b = aq_1 + r_1 & a > r_1 > 0 \\ a = r_1q_2 + r_2 & r_1 > r_2 > 0 \\ r_1 = r_2q_3 + r_3 & r_2 > r_3 > 0 \\ \vdots & \vdots \\ r_{n-2} = r_{n-1}q_n + r_n & r_{n-1} > r_n > 0. \end{array}$$

1.3.1 Los enteros módulo n

Considerando el conjunto de números enteros $\mathbb{Z}$ y definiendo la siguiente relación en $\mathbb{Z}$.

Dado $n \in \mathbb{Z}$ fijo.

$$a \sim b \text{ si y sólo si } n \mid a - b$$

$\sim$ es una relación de equivalencia pués:

1. $\sim$ es reflexiva, si $n \mid 0 = a - a$ implica $a \sim a$.

2. $\sim$ es simétrica, en efecto $a \sim b$ si y sólo si $n \mid a-b$ si y sólo si $n \mid -(a-b) = b-a$ si y sólo si $b \sim a$.

3. $\sim$ es transitiva, si $a \sim b$ y $b \sim c$ entonces $n \mid a-b$ y $n \mid b-c$ implica $n \mid (a-b)+(b-c) = a-c$ luego $a \sim c$.

$\sim$ entonces induce una partición de $\mathbb{Z}$ cuyas clases de equivalencia estan dadas por $\overline{0}$, $\overline{1}, \ldots, \overline{n-1}$.

En donde $\overline{a}$ para $0 \leq a < n$ denota al conjunto de enteros cuyo residuo al dividirlo por n es a. Es decir $\sim$ es equivalente a la relación $a \sim b$ si y sólo si el residuo de dividir a y b por n es el mismo. A saber:

$$\overline{0} = \{\ldots, -n, 0, n, 2n, 3n, \ldots\}$$

$$\overline{1} = \{\ldots, -n+1, 1, n+1, 2n+1, 3n+1, \ldots\}$$

$$\overline{n-1} = \{\ldots, -n-1, -1, n-1, 2n-1, 3n-1, \ldots\}.$$

Este conjunto de clases será denotado por $\mathbb{Z}_n$ y se le llamará el conjunto de enteros módulo n, o el conjunto de clases de equivalencias módulo n.

Se definen en $\mathbb{Z}_n$ las siguientes operaciones:

$$\cdot : \mathbb{Z}_n \times \mathbb{Z}_n \to \mathbb{Z}_n$$

$$+ : \mathbb{Z}_n \times \mathbb{Z}_n \to \mathbb{Z}_n$$

dadas por $\overline{a} \cdot \overline{b} = \overline{ab}$ y $\overline{a} + \overline{b} = \overline{a+b}$ respectivamente.

Note que $\cdot$, $+$ estan bien definidas en $\mathbb{Z}_n$, es decir, no depende del representante de clase, en efecto, suponga $\overline{a} = \overline{a_1}$ y $\overline{b} = \overline{b_1}$ con $a \neq a_1$ y $b \neq b_1$, entonces

$$\overline{a} + \overline{b} = \overline{a+b}$$

$$\overline{a_1} + \overline{b_1} = \overline{a_1+b_1}$$

viendo que $\overline{a+b} = \overline{a_1+b_1}$. En efecto

$$\overline{a} = \overline{a_1} \text{ implica } n \mid a - a_1$$

$$\overline{b} = \overline{b_1} \text{ implica } n \mid b - b_1$$

por lo tanto $n \mid (a - a_1) + (b - b_1) = (a + b) - (a_1 + b_1)$, luego $\overline{a + b} = \overline{a_1 + b_1}$ y análogamente se tiene que

$$\overline{a} = \overline{a_1} \text{ implica } n \mid a - a_1$$

$$\overline{b} = \overline{b_1} \text{ implica } n \mid b - b_1$$

por lo tanto $n \mid b_1(a - a_1)$ y $n \mid a(b - b_1)$, luego $n \mid b_1(a - a_1) + a(b - b_1) = -b_1a_1 + ab$ así que $n \mid ab - b_1a_1$, es decir, $ab \sim b_1a_1$ y $\overline{ab} = \overline{a_1b_1}$.

$$\overline{a_n \cdots a_1a_0} = \overline{0}$$

$$\overline{a_n 10^n + \cdots + a_1 10 + a_0}$$
$$\overline{a_n 10^n} + \cdots + \overline{a_1 10} + \overline{a_0}$$
$$\overline{a_n}\,\overline{10^n} + \cdots + \overline{a_1}\,\overline{10} + \overline{a_0}$$
$$\overline{a_n} \cdot \overline{1^n} + \cdots + \overline{a_1} \cdot \overline{1} + \overline{a_0}$$
$$\overline{a_n} + \cdots + \overline{a_1} + \overline{a_0}.$$

Ejemplo 1.3.6 Suponga $n = 5$ las clases son:

$$\overline{0},\ \overline{1},\ \overline{2},\ \overline{3},\ \overline{4}$$

$$\overline{1} + \overline{3} = \overline{1 + 3} = \overline{4} = \overline{11} + \overline{8} = \overline{19}$$
$$\overline{4} + \overline{3} = \overline{4 + 3} = \overline{7}$$
$$\overline{3} \cdot \overline{2} = \overline{6}$$
$$\overline{0} = \{\ldots, 0, 5, 10, 15, \ldots\}$$
$$\overline{1} = \{\ldots, 1, 6, 11, 16, \ldots\} = \overline{11}$$
$$\overline{2} = \{\ldots, 2, 7, 12, 17, \ldots\} = \overline{17}$$
$$\overline{3} = \{\ldots, 3, 8, 13, 18, \ldots\} = \overline{8}$$
$$\overline{4} = \{\ldots, 4, 9, 14, 19 \ldots\} = \overline{19}.$$

Y tomando un caso más especifico; sea $n = 3$, y sea x el número cifrado $a_n \ldots a_1a_0$ en donde $a_n,\ \ldots,\ a_1,\ a_0$ son dígitos entonces

$$\overline{x} = \overline{a_n 10^n + \cdots + a_1 10 + a_0} =$$

$$\overline{a_n 10^n} + \cdots + \overline{a_1 10} + \overline{a_0} =$$
$$\overline{a_n}\,\overline{10^n} + \cdots + \overline{a_1}\,\overline{10} + \overline{a_0} =$$
$$\overline{a_n} \cdot \overline{1^n} + \cdots + \overline{a_1} \cdot \overline{1} + \overline{a_0} =$$
$$\overline{a_n} + \cdots + \overline{a_1} + \overline{a_0} =$$
$$\overline{a_n + \cdots + a_1 + a_0}$$

de donde se deduce que x es divisible por 3 si y sólo si 3 divide a $a_n + \cdots + a_1 + a_0$.

1.3.2 Grupos

Definición 1.3.7 *Un grupo es un conjunto no vacio junto con una operación binaria $\circ : G \times G \to G$ que satisface:*

1. *$(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$ para todo $g_1,\ g_2,\ g_3 \in G$, es decir, es asociativa.*
2. *Existe $e \in G$ llamado neutro si cumple $g_1 \circ e = e \circ g_1 = g_1$ para todo $g_1 \in G$.*
3. *Dado $g \in G$ existe (inverso) $h \in G$ tal que $g \circ h = h \circ g = e$.*

Algunas propiedades de los puntos (*2*) y (*3*) que se pueden tener es que e es único y dado g también h lo es y por tanto se denota $h = g^{-1}$.
Unicidad de e: Si $e_1 \circ g = g \circ e_1 = g\ \forall\ g \in G$ se tiene en particular

$$e_1 = e \circ e_1 = e_1 \circ e = e$$

Unicidad de g^{-1}: Si $g \circ h = h \circ g = e$ y $g \circ h_1 = h_1 \circ g = e$, entonces, $h_1 = h_1 \circ e = h_1 \circ (g \circ h) = (h_1 \circ g) \circ h = e \circ h = h$ luego $h = h_1$ y se denota por $h = h_1 = g^{-1}$.

Ejemplo 1.3.8

1. Sea $\mathbb{R}$ con $+$ (la suma usual) es un grupo ($\circ = +$)
   (a) $(a + b) + c = a + (b + c)$ para todo $a,\ b,\ c \in \mathbb{R}$.
   (b) existe $0 \in \mathbb{R}$ tal que $a + 0 = 0 + a = a$ para todo $a \in \mathbb{R}$.
   (c) dado $a \in \mathbb{R}$ existe $-a \in \mathbb{R}$ tal que $a + (-a) = (-a) + a = 0$

por lo tanto es un grupo.

2. $(\mathbb{R} \setminus \{0\}, \circ)$ forma un grupo.
3. $(\mathbb{Z}_n, +)$ es un grupo.
4. $(\mathbb{Z}_2, \circ)$ es un grupo con
$$\mathbb{Z}_2 = \{\overline{0}, \overline{1}\}$$
$$\overline{0} \circ \overline{0} = \overline{0}$$
$$\overline{0} \circ \overline{1} = \overline{0}$$
$$\overline{1} \circ \overline{0} = \overline{0}$$
$$\overline{1} \circ \overline{1} = \overline{1}.$$
   (a) $\circ$ es asociativa porque se hereda de la asociatividad de $\mathbb{Z}$.
   (b) el neutro es $\overline{1}$.
   (c) $\overline{0}$ no tiene inverso.
5. $(\mathbb{Z}_4 \setminus \{0\}, \circ)$ no es un grupo
$$\mathbb{Z}_4 = \{\overline{1}, \overline{2}, \overline{3}\}$$
$$\overline{3} \circ \overline{3} = \overline{1}$$
$$\overline{1} \circ \overline{1} = \overline{1}$$
$$\overline{2} \circ \overline{2} = \overline{0}$$
$$\overline{2} \circ \overline{1} = \overline{2} \neq \overline{1}$$
$$\overline{2} \circ \overline{3} = \overline{2}.$$

Proposición 1.3.9 *En general* $(\mathbb{Z}_n \setminus \{\overline{0}\}, \circ)$ *es un grupo si y sólo si* n *es primo.*

Demostración Para probar la necesidad sea $1 \leq a < n$ divisor de n entonces $\overline{a} \in \mathbb{Z}_n \setminus \{0\}$ y existe $1 \leq x < n$ tal que $\overline{xa} = \overline{1}$, luego $n \mid 1 - xa$, es decir, $1 - ax = ny$ con $y \in \mathbb{Z}$ de donde $1 = ny + ax = (as)y + ax = a(sy + x)$ para algún $s \in \mathbb{Z}$, luego $a \mid 1$ y por tanto $a = 1$ así que n es primo.
Para probar la suficiencia suponga ahora $n = p$ primo, viendo que $(\mathbb{Z}_p \setminus \{0\}, \circ)$ sea un grupo

1. La asociatividad se hereda de la asociatividad en $\mathbb{Z}$ y la forma en como se define.

2. Existe $\overline{1} \in \mathbb{Z}_p \setminus \{0\}$ tal que $\overline{a} \circ \overline{1} = \overline{a}$ para todo $a \in \mathbb{Z}_p \setminus \{0\}$.

3. Dado $\overline{a} \in \mathbb{Z}_p \setminus \{0\}$, tomando $\overline{b} = \overline{a}$ tal que $1 \leq b < p-1$ luego $1 = (b, p)$ y por tanto existe $x,\ y \in \mathbb{Z}$ tal que $1 = xb + yp$.
Tomando la clase $\overline{1} = \overline{x}\overline{b}$, más aún por la conmutatividad de $\mathbb{Z}$ se tiene $\overline{x} = \overline{b}^{-1} = \overline{a}^{-1}$, es decir, $\overline{x} = \overline{a}^{-1}$. □

Nota: En la definición 1.3.7, los incisos *2.* y *3.* se pueden debilitar usando:

ii) Existe $e \in G$ tal que $g \circ e = g$ para todo $g \in G$.

iii) Dado $g \in G$ existe $h \in G$ tal que $g \circ h = e$.

2., *3.* implica ii), iii); esta implicación se sigue de manera inmediata.
Para ver el recíproco suponga que se cumple (ii) y (iii) y mostrando el siguiente teorema:

Teorema 1.3.10 *Bajo las hipótesis 1.,* ii) *y* iii) *si $g \circ g = g$ para algún $g \in G$ entonces $g = e$.*

Demostración Sea $g \in G$ tal que $g \circ g = g$ y $g' \in G$ tal que $g \circ g' = e$ entonces $g \circ (g \circ g') = g \circ e = g = g \circ (g \circ g') = (g \circ g) \circ g' = g \circ g' = e$.
Probando ahora que ii) y iii) implican *2.* y *3.*, suponga $g \circ e = g$ y viendo que $e \circ g = g$

$$(h \circ g) \circ (h \circ g) = (h \circ (g \circ h)) \circ g = (h \circ e) \circ g = h \circ g$$

y por la definición 1.3.7 punto *3.* $h \circ g = e$.
Ahora si $g \circ e = g$ entonces $e \circ g = g$.
Dado $g \in G$ existe h tal que $g \circ h = e$ así que

$$e \circ g = (g \circ h) \circ g = g \circ (h \circ g) = g \circ (g \circ h) = g \circ e = g.$$

□

Proposición 1.3.11 *Si $g,\ g_1 \in G$ y G es un grupo entonces:*

1. $(g^{-1})^{-1} = g$.

2. $(g \circ g_1)^{-1} = g_1^{-1} \circ g^{-1}$.

Demostración Para *1.* se tiene que $g^{-1} \circ (g^{-1})^{-1} = e$ así que

$$g = g \circ e = g \circ (g^{-1} \circ (g^{-1})^{-1}) = (g \circ g^{-1}) \circ (g^{-1})^{-1} = e \circ (g^{-1})^{-1} = (g^{-1})^{-1}$$

Para *2.* se tiene que

$$(g_1^{-1} \circ g^{-1}) \circ (g \circ g_1) = g_1^{-1} \circ (g^{-1} \circ g) \circ g_1 =$$

$$g_1^{-1} \circ (e \circ g_1) = g_1^{-1} \circ g_1 = e$$

y por la unicidad del inverso $(g \circ g_1)^{-1} = g_1^{-1} \circ g^{-1}$. □

1.3.3 Subgrupos

Definición 1.3.12 *Sea* $(G, \circ)$ *un grupo y* $H \subseteq G$ *tal que* $(H, \circ)$ *es un grupo, diremos que* H *es subgrupo de* G *y se denotará por* $H \leq G$.

Ejemplo 1.3.13

1. $(\mathbb{Z}, +)$ es un grupo. Sea $H = \{\ldots, -4, -2, 0, 2, 4, \ldots\} = \mathbb{Z}(2)$ entonces $H \leq \mathbb{Z}$.
   Observación: Si $(G, \circ)$ es un grupo, $e \in G$ es el neutro en G y $H \leq G$ entonces $e \in H$ es el neutro de H.

2. Sea $G = \mathbb{Z}_2 \times \mathbb{Z}_2$ con la operación

   $$(\overline{a}, \overline{b}) \circ (\overline{c}, \overline{d}) = (\overline{a} + \overline{c}, \overline{b} + \overline{d})$$

   $(G, \circ)$ es un grupo, calculando cuales son sus subgrupos se tiene:

   $$G = \{(\overline{0}, \overline{0}), (\overline{0}, \overline{1}), (\overline{1}, \overline{0}), (\overline{1}, \overline{1})\}$$

   $$H_0 = \{(\overline{0}, \overline{0})\}$$
   $$H_1 = \{(\overline{0}, \overline{0}), (\overline{0}, \overline{1})\}$$
   $$H_2 = \{(\overline{0}, \overline{0}), (\overline{1}, \overline{0})\}$$
   $$H_3 = \{(\overline{0}, \overline{0}), (\overline{1}, \overline{1})\}$$
   $$H_4 = G.$$

Teorema 1.3.14 *Sea $(G, \circ)$ grupo y $H \subseteq G$ entonces las siguientes condiciones son equivalentes*

1. $H \leq G$.

2. *para todo* $h_1, h_2 \in H$; $h_1 \circ h_2^{-1} \in H$.

3. *para todo* $h_1, h_2 \in H$

   3.1 $h_1 \circ h_2 \in H$.

   3.2 $h_1^{-1} \in H$.

Demostración *1.* implica *2.*. Como $H \leq G$ entonces para $h_2 \in H$ se tiene $h_2^{-1} \in H$ (por *3.* de la definición 1.3.7) y como $\circ$ debe ser operación en H. $h_1 \circ h_2^{-1} \in H$. *2.* implica *3.*. Para demostrar *3.2*, sean h_1, $h_2 \in H$ entonces $h = h_1 \circ h_2^{-1}$, tomando $h_1 = h_2$ se tiene $e = h_1 \circ h_1^{-1} \in H$, luego para h_1, $e \in H$ se tiene $e \circ h_1^{-1} = h_1^{-1} \in H$. Para demostrar *3.1*, sean h_1, $h_2 \in H$ por *3.2* $h_2^{-1} \in H$ luego $h_1 \circ (h_2^{-1})^{-1} \in H$, entonces $h_1 \circ h_2 \in H$, por lo cual *3.* implica *1.*. Ahora viendo que H sea un grupo y de las propiedades de los grupos se tiene que la asociatividad se hereda de la asociatividad en G, $e \in H$ porque $e = h_1 \circ h_1^{-1} \in H$, H tiene inverso dado el punto *3.2*. □

Definición 1.3.15 *Sea $(G, \circ)$ un grupo. Si $g_1 \circ g_2 = g_2 \circ g_1$ para todo g_1, $g_2 \in G$ el grupo se dice conmutativo o abeliano.*

Ejemplo 1.3.16

1. $(\mathbb{Z}_n, +)$ es abeliano.

2. $(\mathbb{Z}_p \backslash \{0\}, \circ)$ es abeliano (p-primo).

3. Sea $X = \{1, 2, 3\}$, S_X el grupo de permutaciones de X, $\sigma \in S_X$:

$$\begin{array}{lll}
\sigma_0 : X \to X & & \\
\quad 1 \mapsto 1 & & \\
\quad 2 \mapsto 2 & \binom{123}{123} & \mapsto (1) \\
\quad 3 \mapsto 3 & & \\
\sigma_1 : X \to X & & \\
\quad 1 \mapsto 2 & & \\
\quad 2 \mapsto 1 & \binom{123}{213} & \mapsto (12) \\
\quad 3 \mapsto 3 & & \\
\sigma_2 : X \to X & & \\
\quad 1 \mapsto 3 & & \\
\quad 2 \mapsto 2 & \binom{123}{321} & \mapsto (13) \\
\quad 3 \mapsto 1 & & \\
\sigma_3 : X \to X & & \\
\quad 1 \mapsto 1 & & \\
\quad 2 \mapsto 3 & \binom{123}{132} & \mapsto (23) \\
\quad 3 \mapsto 2 & &
\end{array}$$

$$\begin{array}{lll}
\sigma_4 : X \to X & & \\
\quad 1 \mapsto 2 & & \\
\quad 2 \mapsto 3 & \binom{123}{231} & \mapsto (123) \\
\quad 3 \mapsto 1 & & \\
\sigma_5 : X \to X & & \\
\quad 1 \mapsto 3 & & \\
\quad 3 \mapsto 2 & \binom{123}{312} & \mapsto (132) \\
\quad 2 \mapsto 1 & &
\end{array}$$

no es conmutativo ya que $\sigma_2 \circ \sigma_3 = (213)$ y $\sigma_3 \circ \sigma_2 = (123)$ luego, $\sigma_5 = \sigma_2 \circ \sigma_3 \neq \sigma_3 \circ \sigma_2 = \sigma_4$.

Teorema 1.3.17 *Sea $(G, \circ)$ un grupo, entonces la intersección de subgrupos de G es un subgrupo.*

Demostración Sea $K = \bigcap_{\substack{H_\lambda \leq G \\ \lambda \in I}} H_\lambda$, I un conjunto arbitrario de índices y sean $h_1, h_2 \in K$ entonces, $h_1, h_2 \in H_\lambda$ para todo $\lambda \in I$, de donde $h_1 \circ h_2^{-1} \in H_\lambda$ para todo $\lambda \in I$, luego $h_1 \circ h_2^{-1} \in K$ de donde $K \leq G$. □

Observación: La unión de subgrupos de un grupo G no necesariamente es un subgrupo.

Ejemplo 1.3.18 Sea $G = S_3$, $H_1 \cup H_2 = \{e, \sigma_1, \sigma_2\} \nleq G$.

Definición 1.3.19 *Sea G un grupo $S \subseteq G$ un subconjunto, se denota por $< s >$ al conjunto*

$$< s >= \{g_1^{a_1} \circ \cdots \circ g_n^{a_n} : a_i \in \mathbb{Z}, g_i \in S \text{ para todo } i \text{ con } n \in \mathbb{N}\}$$

$< s > \leq G$ llamado el subgrupo generado por S. Sean x, $y \in < s >$ entonces $x = g_1^{a_1} \circ \cdots \circ g_n^{a_n}$, $y = h_1^{b_1} \circ \cdots \circ h_m^{b_m}$ con g_i, $h_j \in S$, a_i, $b_j \in \mathbb{Z}$ para todo $1 \leq i \leq n$, $1 \leq j \leq m$ entonces $y^{-1} = h_m^{-b_m} \circ \cdots \circ h_0^{-b_1}$ y $xy^{-1} = g_1^{a_1} \circ \cdots \circ g_n^{a_n} h_m^{-b_m} \circ \cdots \circ h_0^{-b_1} \in < s >$, mas aún $< s >$ es el mínimo subgrupo de G que contiene a S. La minimalidad es en el siguiente sentido:

$$< s >= \bigcap_{\substack{H \leq G \\ S \subseteq H}} H$$

$H \leq G$ y $S \subseteq H$.

Definición 1.3.20 *Sean $(G, \circ)$ y $(G_1, *)$ dos grupos.*
Un homomorfismo de G en G_1 (homomorfismo de grupo) es una función $f : G \to G_1$ tal que

$$f(g \circ g_1) = f(g) * f(g_1) \text{ para todo } g,\ g_1 \in G.$$

Si f es además inyectiva, se dice monomorfismo (de grupos).
Si f es suprayectiva, se dice epimorfismo (de grupos).
Si f es biyectiva se dice isomorfismo.

Ejemplo 1.3.21

1. Sea $T : V \to W$ una transformación lineal, entonces al considerar los grupos (V, T_1), (W, T_2), T es un homomorfismo de grupos.

2. Sea $T : \mathbb{R}^3 \to \mathbb{R}^2$ dada por $T(x, y, z) = (x, y)$, entonces T es un epimorfismo de $(\mathbb{R}^3, T_1)$ a $(\mathbb{R}^2, T_2)$. En efecto:

$$T((x_1, y_1, z_1) + (x_2, y_2, z_2)) = T(x_1 + x_2, y_1 + y_2, z_1 + z_2) =$$

$$(x_1 + x_2, y_1 + y_2) = (x_1, y_1) + (x_2, y_2) = T(x_1, y_1, z_1) + T(x_2, y_2, z_2)$$

luego T es homomorfismo y como es sobre es epimorfismo.

Notación: Si G y G_1 son dos grupos isomorfos se denotará por $G \cong G_1$.

Definición 1.3.22 *Sea $\varphi : G \to G_1$ un homomorfismo de grupos, el kernel de φ denotado por* $\ker(\varphi)$ *es el conjunto*

$$\ker(\varphi) = \{g \in G : \varphi(g) = e_{G_1}\}$$

y la imagen es

$$\mathrm{Im}(\varphi) = \{g_1 \in G_1 \colon\ g \in G \text{ con } \varphi(g) = g_1\}.$$

Definición 1.3.23 *Sea $(G, \circ)$ un grupo se dice que es finitamente generado si existe $g_1, \ldots, g_n \in G$ tales que $< g_1, \ldots, g_n >= G$.*

Ejemplo 1.3.24

1. $S_3 =< \sigma_1, \sigma_2 >$ es finitamente generado.
2. $(\mathbb{Z}, +) =< 1 >$ es finitamente generado.

Observación: Si G es finito entonces es finitamente generado.

Definición 1.3.25 *Sea G un grupo $g \in G$ el orden de g denotado por $o(g)$ o por $|g|$ es el mínimo entero positivo (si existe) tal que $g^{o(g)} = e$.*
Si dicho entero no existe se entenderá que $o(g) = +\infty$.

Ejemplo 1.3.26

1. En $(S_3, \circ)$, $o(\sigma_1) = 2$, $\sigma_1 = (12)$; en efecto:
$$\sigma_1^1 = \sigma_1 = (12)$$
$$\sigma_1^2 = \sigma_1 \circ \sigma_1 = (12)(12) = (1) = e = \sigma_0.$$
2. Para $(\mathbb{Z}, +)$, $o(1) = +\infty$, $1^n = n$; en efecto:
$$1^0 = 1$$
$$1^2 = 1 \circ 1 = 1 + 1 = 2$$
$$1^3 = 1 \circ 1 \circ 1 = 1 + 1 + 1 = 3$$
$$\vdots$$
$$1^n = \underbrace{1 \circ 1 \circ 1 \circ \cdots \circ 1}_{n \text{ veces}} = \underbrace{1 + 1 + 1 + \cdots + 1}_{n \text{ veces}} = n$$

   Note que:

(a) $o(e) = 1$.

(b) Si $(G, \circ)$ es finito, $o(g) < +\infty$ para todo $g \in G$.
Demostración Sea $G = \{g_0 = e, g_1, \ldots, g_n\}$. Suponga que $o(g)$ no es finito, entonces $g^k \neq g^l$, si $k \neq l$ pues $g^k = g^l$ implica $g^{k-l} = e$ con lo que se tendría $o(g) < +\infty$, luego $g,\ g^2,\ \ldots,\ g^n,\ g^{n+1},\ g^{n+2} \in G$ y son distintos lo cual es una contradicción por lo que (b) se cumple. □

Definición 1.3.27 *Sea $(G, \circ)$ un grupo G se dice cíclico si $G =< g >$ para algún $g \in G$.*

Ejemplo 1.3.28

1. $(\mathbb{Z}, +)$ es cíclico (abeliano), $\mathbb{Z} =< 1 >$.
2. $(\mathbb{Z}_n, +)$ es cíclico (abeliano), $\overline{1}+\overline{1} = \overline{2}$, $(3)(\overline{1}) = \overline{3}$, $(4)(\overline{1}) = \overline{4}$, $(n)(\overline{1}) = \overline{n}$.
3. S_3 no es cíclico (no es abeliano).
4. $(\mathbb{Z}_2 \times \mathbb{Z}_2, +)$ no es cíclico (es abeliano).

Teorema 1.3.29 *Sean $H,\ K \leq G$, entonces $KH < G$ si y sólo si $HK = KH$.*

Demostración Para probar la necesidad, tomando $h_1 \in H$, $k_1 \in K$; como $HK \leq G$, entonces $(h_1k_1)^{-1} \in HK$ así que existe $k \in K$, $h \in H$ tal que $(h_1k_1)^{-1} = hk$, por tanto $k^{-1}h^{-1} = (hk)^{-1} = ((h_1k_1)^{-1})^{-1} = h_1k_1$ y por ser $K,\ H \leq G$ se tiene $k^{-1} \in K$, $h^{-1} \in H$, luego $h_1k_1 \in KH$ de donde $HK \subseteq KH$. Por simetría $HK \supseteq KH$ y $HK = KH$.
Para probar la suficiencia, sea g_1, $g_2 \in HK$ entonces $g_1 = h_1k_1$, $g_2 = h_2k_2$ para algunos h_1, $h_2 \in H$, k_1,$k_2 \in K$ luego $g_1g_2 = h_1k_1h_2k_2$ pero $k_1h_2 \in KH = HK$, por tanto $k_1h_2 = hk$ con $h \in H$, $k \in K$,de donde $g_1g_2 = h_1hkk_2 \in HK$, mas aún, $g_1^{-1} = k_1^{-1}h_1^{-1} \in KH = HK$. □

Teorema 1.3.30 *Sea $H \leq G$, G grupo entonces $Ha = Hb$ si y sólo si $ab^{-1} \in H$.*

Demostración Para probar la necesidad, sea $ea \in Ha = Hb$ implica $ea = hb$, con $h \in H$, luego $ab^{-1} = h \in H$.
Para probar la suficiencia, note que $ab^{-1} \in H$ implica $(ab^{-1})^{-1} = ba^{-1} \in H$ así que basta probar una contención por simetría. Tome $g \in Ha$ entonces $g = ha$ con $h \in H$, y como $ab^{-1} \in H$ entonces $hab^{-1} = h_1 \in H$, luego $g = ha = h_1b \in Hb$ de donde $Ha \subseteq Hb$. □

Definición 1.3.31 *Sea $(G, \circ)$ un grupo, $g \in G$, se le denomina clase lateral derecha a*

$$Hg = \{hg : h \in H\}$$

y clase lateral izquierda a

$$gH = \{gh : h \in H\}.$$

Teorema 1.3.32 *Sea G grupo, el conjunto de clases laterales (izquierdas o derechas) forman una partición de G.*

Demostración Note que $G = \bigcup_{g \in G} Hg$ claramente pues $e \in H$ por lo que basta ver que las clases son disjuntas, suponga

$$Hg_1 \cap Hg_2 \neq \phi$$

y sea $a \in Hg_1 \cap Hg_2$ entonces $a = h_1 g_1 = h_2 g_2$ con h_1, $h_2 \in H$ luego $g_1 g_2^{-1} = h_1^{-1} h_2 \in H$ así que por el teorema 1.3.30 $Hg_1 = Hg_2$. □
Notación: Sea $(G, \circ)$ grupo, $H \leq G$; $\mathfrak{R}_H$ denota el conjunto de clases laterales derechas y $\mathfrak{L}_H$ al conjunto de clases laterales izquierdas.

Teorema 1.3.33 *Sea $(G, \circ)$ grupo, $H \leq G$ entonces $|\mathfrak{R}_H| = |\mathfrak{L}_H|$.*

Demostración Sea $f : \mathfrak{R}_H \to \mathfrak{L}_H$ dada por $Ha \to a^{-1}H$ viendo que f está bién definida (no depende del representante de clase). Suponga que $Ha = Hb$ entonces $ab^{-1} \in H$ luego $(ab^{-1})^{-1} = ba^{-1} \in H$ implica $a^{-1}H = b^{-1}H$.
Para ver que f es inyectiva se tiene que $f(Ha) = f(Hb)$ si y sólo si $a^{-1}H = b^{-1}H$ si y sólo si $ba^{-1} \in H$ ai y sólo si $(ba^{-1})^{-1} = ab^{-1} \in H$ si y sólo si $Ha = Hb$.
Para ver que f es sobre, sea $aH \in \mathfrak{L}_H$ entonces $a^{-1}H \in \mathfrak{L}_H$ y $f(Ha^{-1}) = (a^{-1})^{-1}H = aH$, luego f es biyectiva y $|\mathfrak{R}_H| = |\mathfrak{L}_H|$. □
Como $|\mathfrak{R}_H| = |\mathfrak{L}_H|$ a este valor se le denomina el índice de H en G y se denota por $[G : H]$.

Teorema 1.3.34 *[Índice de Lagrange] Sea $(G, \circ)$ grupo $H \leq G$ entonces*

$$|G| = [G : H]|H|.$$

Demostración Si $[G : H] = +\infty$, entonces $|G| = +\infty$ y se tiene la igualdad. Si $|H| = +\infty$ entonces $|G| = +\infty$ y se tiene la igualdad. Note que $G = \dot{\bigcup}_{g \in G} Hg$ (la unión anterior representa una unión disjunta) así que

$$|G| = \Big| \dot{\bigcup}_{g \in G} \Big| = \sum_{\substack{[G:H] \\ g \in G}} |Hg| = \sum_{\substack{[G:H] \\ g \in G}} |H| = [G : H]|H|.$$

□

Corolario 1.3.35 *Sea G un grupo finito, $g \in G$ entonces $o(g) \mid |G|$, en particular*

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

Demostración Sea $H =< g >$ (el subgrupo cíclico de G generado por g), por el teorema 1.3.34

$$|G| = [G : H]|H|$$

pero $o(g) = |H|$, así $o(g) \mid |G|$ y $g^{|G|} = g^{[G:H]o(g)} = \left(g^{o(g)}\right)^{[G:H]} = e^{[G:H]} = e$. □

Capítulo 2

Congruencias

En este capítulo se introducen los conceptos fundamentales de congruencia, así como sus operaciones y algunos resultados inmediatos que se cumplen. Esto permitirá entender bién las notaciones y operaciones que se realizarán en el capítulo tres. En particular se realiza una prueba básica del teorema chico de Fermat.

2.1 Conceptos fundamentales

Definición 2.1.1 *Sea $\mathbb{Z}$ el conjunto de números enteros y $m \in \mathbb{Z}\backslash\{0\}$ fijo. Diremos que a, $b \in \mathbb{Z}$ son congruentes módulo m si el residuo de dividir a por m es el mismo que el de dividir b por m.*
La congruencia de los números a y b respecto del módulo m se denota por:

$$a \equiv b \mod m$$

y se lee: a es congruente con b respecto del módulo m.

La congruencia de los números a y b respecto del módulo m es equivalente a:

Proposición 2.1.2

1. *La posibilidad de expresar a en la forma $a = b + mt$, donde t es entero.*

2. *La divisibilidad de $a - b$ por m.*

Demostración En efecto, dado que $a \equiv b \mod m$ se deduce que

$$a = mq + r,\ b = mq_1 + r;\ 0 \leq r < m,$$

de donde

$$a - b = m(q - q_1),\ a = b + mt,\ t = q - q_1.$$

Análogamente, de $a = b + mt$, representando b en la forma

$$b = mq_1 + r,\ 0 \leq r < m$$

se deduce que

$$a = mq + r;\ q = q_1 + t$$

es decir,

$$a \equiv b \mod m.$$

Por esto la definición 2.1.1 se cumple. □

2.1.1 Propiedades de las congruencias

Proposición 2.1.3

a) *Dos números que son congruentes con un tercero, son congruentes entre sí.*

b) *Las congruencias se pueden sumar término a término.*

c) *Las congruencias se pueden multiplicar término a término.*

d) *Las propiedades* **b)** *y* **c)** *(la adición y multiplicación de congruencias) se generalizan mediante lo siguiente: si en la expresión de una función racional entera de coeficientes enteros*

$$S = \sum A_{\alpha_1,\ldots,\alpha_k} x_1^{\alpha_1} \cdots x_k^{\alpha_k}$$

se sustituyen los números $A_{\alpha_1,\ldots,\alpha_k}$, $x_1, \ldots, x_k$ por los números $B_{\alpha_1,\ldots,\alpha_k}$, $y_1, \ldots, y_k$, los cuales son congruentes con los anteriores respecto del módulo m, la expresión nueva de S será congruente con la expresión anterior respecto del módulo m.

e) *Ambos miembros de la congruencia se pueden dividir por su común divisor, si este último es primo con el módulo.*

Demostración

a) La primera afirmación se deduce de la definición 2.1.1 y la proposición 2.1.2.

b) Para la segunda sea

$$a_1 \equiv b_1 \mod m,\ a_2 \equiv b_2 \mod m,\ \ldots, a_k \equiv b_k \mod m \tag{2.1}$$

entonces,

$$a_1 = b_1 + mt_1,\ a_2 = b_2 + mt_2, \ldots, a_k = b_k + mt_k \tag{2.2}$$

de donde

$$a_1 + a_2 + \ldots + a_k = b_1 + b_2 + \ldots + b_k + m(t_1 + t_2 + \ldots + t_k)$$

o sea

$$a_1 + a_2 + \ldots + a_k \equiv b_1 + b_2 + \ldots + b_k \mod m$$

en particular se tiene que $a + b \equiv c \mod m$ implica $a \equiv c - b \mod m$ y que como $mk \equiv 0 \mod m$ entonces $a + mk \equiv b \mod m$.

c) Para la tercera afirmación, examinando de nuevo las congruencias (2.1) y las igualdades (2.2) que se deducen de ellas. Multiplicando término a término las igualdades (2.2), se tiene que

$$a_1 a_2 \ldots a_k = b_1 b_2 \ldots b_k + mN,$$

donde N es entero. Por consiguiente,

$$a_1 a_2 \ldots a_k = b_1 b_2 \ldots b_k \mod m$$

En particular ambos miembros de la congruencia se pueden elevar a una misma potencia. Además ambos miembros de la congruencia se pueden multiplicar por un mismo entero. En efecto, multiplicando la congruencia $a \equiv b \mod m$ por la congruencia evidente $k \equiv k \mod m$, se obtiene $ak \equiv bk \mod m$.

d) Para la cuarta afirmación se tiene que en efecto, de

$$A_{\alpha_1,\ldots,\alpha_k} \equiv B_{\alpha_1,\ldots,\alpha_k} \mod m$$

$$x_1 \equiv y_1 \mod m, \ldots, x_k \equiv y_k \mod m$$

hallando **c)**

$$x_1^{\alpha_1} \equiv y_1^{\alpha_1} \mod m, \ldots, x_k^{\alpha_k} \equiv y_k^{\alpha_k} \mod m$$

$$A_{\alpha_1,\ldots,\alpha_k} x_1^{\alpha_1} \cdots x_k^{\alpha_k} \equiv B_{\alpha_1,\ldots,\alpha_k} y_1^{\alpha_1} \cdots y_k^{\alpha_k} \mod m$$

de donde, sumando, se tiene

$$\sum A_{\alpha_1,\ldots,\alpha_k} x_1^{\alpha_1} \cdots x_k^{\alpha_k} \equiv \sum B_{\alpha_1,\ldots,\alpha_k} y_1^{\alpha_1} \cdots y_k^{\alpha_k} \mod m.$$

En particular si

$$a \equiv b \mod m,\ a_1 \equiv b_1 \mod m, \ldots, a_n \equiv b_n \mod m$$

$$x \equiv x_1 \mod m$$

se tiene

$$ax^n + a_1 x^{n-1} + \ldots + a_n \equiv bx_1^n + b_1 x_1^{n-1} + \ldots + b_n \mod m.$$

e) Para la quinta afirmación se tiene que, si $a \equiv b \mod m$, $a = a_1 d$, $b = b_1 d$, $(d,\ m) = 1$ resulta que la diferencia $a - b$, igual a $(a_1 - b_1)d$, es divisible por m. Por esto $a_1 - b_1$ es divisible por m, es decir, $a_1 \equiv b_1 \mod m$. □

2.1.2 Otras propiedades de las congruencias

Proposición 2.1.4

a) *Ambos miembros de una congruencia y el módulo se pueden multiplicar por un mismo número entero.*

b) *Ambos miembros de una congruencia y el módulo se pueden dividir por cualquier común divisor suyo.*

c) *Si se verifica la congruencia $a \equiv b$ respecto de varios módulos, entonces se verifica también respecto del módulo que es igual al mínimo común múltiplo de estos módulos.*

d) *Si una congruencia se verifica respecto de un módulo m, también se verifica respecto de un módulo d que sea igual a cualquier divisor del número m.*

e) *Si un miembro de una congruencia y el módulo son divisibles por algún número, el otro miembro de la congruencia tiene que ser divisible por el mismo número.*

f) *Si $a \equiv b \mod m$, entonces $(a,\ m) = (b,\ m)$.*

Demostración

a) Para la primera afirmación se sabe que de $a \equiv b \mod m$ se deduce que

$$a = b + mt,\ ak = bk + mkt$$

y, por consiguiente, $ak \equiv bk \mod mk$.

b) Para la segunda afirmación, sean a, b y d tales que

$$a \equiv b \mod m,\ a = a_1 d,\ b = b_1 d,\ m = m_1 d.$$

Se tiene que

$$a = b + mt,\ a_1 d = b_1 d + m_1 dt$$

así que

$$a_1 = b_1 + m_1 t$$

y por lo tanto, $a_1 \equiv b_1 \mod m$.

c) Para la tercera afirmación se sabe que de

$$a \equiv b \mod m_1,\ a \equiv b \mod m_2, \ldots,\ a \equiv b \mod m_k$$

se deduce que la diferencia $a - b$ es divisible por todos los módulos $m_1, m_2, \ldots,\ m_k$. Por esto, también es divisible esta diferencia por el mínimo común múltiplo m de estos módulos, es decir, $a \equiv b \mod m$.

d) Para la cuarta afirmación se tiene que en efecto, de $a \equiv b \mod m$ se deduce que la diferencia $a - b$ tiene que ser divisible por m; por esto, esta diferencia tiene que ser divisible también por cualquier divisor d del número m, es decir, $a \equiv b \mod d$.

e) Para la quinta afirmación se tiene que de $a \equiv b \mod m$ se deduce que $a = b + mt$; si a y m son múltiplos de d, entonces también b tiene que ser múltiplo de d, como se afirmaba.

f) Finalmente para la sexta afirmación la igualdad se deduce inmediatamente de $a = b + mt$. □

2.1.3 Sistema completo de restos

Definición 2.1.5 *Los números que dan un mismo resto, o lo que es lo mismo, los que son congruentes respecto del módulo m, forman una clase de números respecto del módulo m.*

De esta definición se deduce que a todos los números de una clase les corresponde un mismo resto r, por lo cual, haciendo recorrer a q en la forma $mq + r$ todos los números enteros, se obtienen todos los números de clase.
Correspondientemente a m valores distintos de r, se tienen m clases de números respecto del módulo m.

Definición 2.1.6 *Cualquier número de la clase se llama resto o residuo respecto del módulo m con relación a todos los números de la misma clase.*

El resto que se obtiene para $q = 0$, igual al residuo mismo r, se llama resto no negativo mínimo.
El resto ρ que es el menor en valor absoluto, se llama resto absoluto mínimo. Evidentemente, si $r < \frac{m}{2}$ se tiene $\rho = r$; si $r > \frac{m}{2}$ se tiene que $\rho = r - m$; finalmente, si m es par y $r = \frac{m}{2}$, se puede tomar por ρ cualquiera de los dos números $\frac{m}{2}$ y $\frac{m}{2} - m = -\frac{m}{2}$.
Tomando un resto de cada clase se obtiene un sistema completo de restos respecto del módulo m. Por lo general, como sistema completo de restos se emplean los restos no negativos mínimos $0, 1, \ldots, m - 1$ o también los

restos absolutos mínimos; como se deduce de lo expuesto anteriormente, estos últimos, en caso de m impar, se representan por la sucesión

$$-\frac{m-1}{2}, \ldots, \ -1, \ 0, \ 1, \ldots, \ \frac{m-1}{2}$$

y en el caso de m par, por cualquiera de las dos sucesiones siguientes

$$-\frac{m}{2}+1, \ldots, \ -1, \ 0, \ 1, \ldots, \ \frac{m}{2}$$

$$-\frac{m}{2}, \ldots, \ -1, \ 0, \ 1, \ldots, \ \frac{m}{2}-1$$

Definición 2.1.7 *Se dice que un conjunto de m números es un sistema completo de restos si son incongruentes dos a dos.*

Proposición 2.1.8 *Si* $(a, \ m) = 1$ *y* x *recorre el sistema completo de restos respecto del módulo* m*, entonces* $ax + b$*, donde* b *es un entero cualquiera, también recorre el sistema completo de restos respecto del módulo* m*.*

Demostración En efecto, hay tantos números de la forma $ax + b$ como números x hay, es decir, m. Según la definición 2.1.7, no queda más que mostrar que dos números cualesquiera $ax_1 + b$ y $ax_2 + b$, que corresponden a dos números incongruentes x_1 y x_2, son también incongruentes entre sí respecto del módulo m.
Pero suponiendo que $ax_1 + b \equiv ax_2 + b \mod m$, se obtiene la congruencia $ax_1 \equiv ax_2 \mod m$, de donde, en virtud de que $(a, \ m) = 1$, resulta $x_1 \equiv x_2 \mod m$, lo cual contradice por completo a la incongruencia de los números x_1 y x_2. □

2.1.4 Sistema reducido de restos

Proposición 2.1.9 *Los números de una misma clase respecto del módulo* m *tienen con el módulo un mismo máximo común divisor.*

Son de suma importancia las clases para las cuales este divisor es igual a la unidad, es decir, las clases que contienen números que son primos con el módulo, ya que estos números van a ser de gran utilidad en la elaboración del programa que se expondrá en esta tesis.

Definición 2.1.10 *Sea $m \in \mathbb{N}\setminus\{0\}$ fijo se define la función de Euler $\varphi : \mathbb{N} \to \mathbb{N}$ como la cantidad de enteros positivos a que son primos relativos con m.*

Ejemplo 2.1.11

$$\varphi(2) = 1,\ \varphi(3) = 2,\ \varphi(4) = 2,\ \varphi(6) = 2.$$

Tomando los restos correspondientes en estas clases, se obtiene el sistema reducido de restos respecto del módulo m. Por consiguiente, el sistema reducido de restos se puede formar de los números del sistema completo que son primos con el módulo. Ordinariamente, el sistema reducido de restos se extrae del sistema de restos no negativos mínimos: $0,\ 1, \ldots,\ m-1$. Como entre éstos hay $\varphi(m)$ números que son primos con m, la cantidad de números del sistema reducido, así como la cantidad de clases que contienen números primos con el módulo, es igual a $\varphi(m)$.

Ejemplo 2.1.12 El sistema reducido de restos módulo 42 es

$$1,\ 5,\ 11,\ 13,\ 17,\ 19,\ 23,\ 25,\ 29,\ 31,\ 37,\ 41.$$

Observación 2.1.13 *Cualesquiera $\varphi(m)$ números que sean incongruentes dos a dos módulo m y que sean primos con el módulo, forman un sistema reducido de restos módulo m.*

Demostración En efecto, estos números, siendo incongruentes dos a dos y primos con el módulo, tienen que pertenecer a distintas clases que contienen números que son primos con el módulo, y como en total hay $\varphi(m)$ de tales números, es decir, tantos cuantas clases hay del tipo indicado, en cada una de las clases habrá, indispensablemente, un número único. □

Proposición 2.1.14 *Si $(a,\ m) = 1$ y x recorre el sistema reducido de restos según el módulo m, ax también recorre el sistema reducido de restos según el módulo m.*

Demostración En efecto, hay tantos números ax como números x hay, es decir, $\varphi(m)$. Por lo tanto, en virtud de la observación 2.1.13, no queda más que demostrar que los números ax son incongruentes dos a dos respecto del módulo m y son primos con el módulo:

Pero lo primero se demostró anteriormente para números de la forma más general $ax + b$; lo segundo se deduce de que $(a,\ m) = 1$ y $(x,\ m) = 1$. □

2.1.5 Teoremas de Euler y Fermat

Teorema 2.1.15 *[Teorema de Euler] Si* $m > 1$ *y* $(a,\ m) = 1$ *se tiene:*

$$a^{\varphi(m)} \equiv 1 \mod m.$$

Demostración En efecto, si x recorre el sistema reducido de restos

$$x = r_1,\ r_2, \ldots,\ r_c;\ c = \varphi(m)$$

formado por los restos no negativos mínimos, entonces los restos no negativos mínimos $\rho_1,\ \rho_2, \ldots,\ \rho_c$ de los números ax también recorren el mismo sistema, pero, generalmente, dispuestos en otro orden.

Multiplicando término a término las congruencias

$$ar_1 \equiv \rho_1 \mod m,\ ar_2 \equiv \rho_2 \mod m,\ \ldots,\ ar_c \equiv \rho_c \mod m$$

se tiene que

$$a^c r_1 r_2 \ldots r_c \equiv \rho_1 \rho_2 \ldots \rho_c \mod m$$

de donde, dividiendo ambos miembros por el producto $r_1 r_2 \ldots r_c = \rho_1 \rho_2 \ldots \rho_c$, resulta que

$$a^c \equiv 1 \mod m.$$

□

Teorema 2.1.16 *[Teorema Pequeño de Fermat] Si* p *es primo y* a *no es divisible por* p*, se tiene:*

$$a^{p-1} \equiv 1 \mod p. \tag{2.3}$$

Demostración Este teorema es una consecuencia del Teorema de Euler para $m = p$.
Al teorema de Fermat se le puede dar una forma más cómoda, si se multiplican ambos miembros de la congruencia (2.3) por a, se obtiene la congruencia

$$a^p \equiv a \mod p$$

la cual es válida ya para todos los valores enteros de a, puesto que también es válida si a es múltiplo de p. □

2.2 Congruencias con una incógnita

2.2.1 Conceptos fundamentales

En esta sección el objetivo principal es el estudio de las congruencias de la siguiente forma general:

$$f(x) \equiv 0 \mod m;\ f(x) = a_0x^n + a_1x^{n-1} + \ldots + a_n.$$

Si a no es divisble por m, el número n se llama grado de la congruencia.

Resolver la congruencia, significa hallar todos los valores de x que la satisfacen. Dos congruencias, a las que satisfacen unos mismos valores de x, se llaman equivalentes. Si a la congruencia (2.4) la satisface algún $x = x_1$, entonces a la misma congruencia la satisfacen también todos los números que son congruentes con x_1 respecto del módulo m : $x \equiv x_1 \mod m$. Toda esta clase de números se considera como una solución. Por lo tanto, la congruencia (2.4) tendrá tantas soluciones cuantos restos del sistema completo la satisfagan.

Ejemplo 2.2.1 A la congruencia

$$x^5 + x + 1 = 0 \mod 7$$

entre los números 0, 1, 2, 3, 4, 5, 6 del sistema completo de restos respecto del módulo 7, la satisfacen dos números: $x = 2$ y $x = 4$. Por ello, la congruencia indicada tiene dos soluciones:

$$x \equiv 2 \mod 7,\ x \equiv 4 \mod 7.$$

2.2.2 Congruencias de primer grado

La congruencia de primer grado, después de trasladar el término independiente (con el signo contrario) al segundo miembro, se ruduce a la forma:

$$ax \equiv b \mod m. \tag{2.4}$$

Comenzando a estudiar el problema del número de soluciones de la congruencia (2.4), tomando primero el caso $(a,\ m) = 1$. En virtud de lo obtenido en la sección anterior, la congruencia considerada admite tantas soluciones cuantos

restos del sistema completo la satisfacen. Más aún, cuando x recorre el sistema completo de restos respecto del módulo m, ax recorre el sistema completo de restos. Por consiguiente, para un valor de x tomado del sistema completo, y sólo para uno, ax será congruente con b. Así pues, si $(a,\ m) = 1$ la congruencia (2.4) admite una única solución.

Supongamos ahora que $(a,\ m) = d > 1$. Entonces, para que la congruencia (2.4) tenga solución es necesario que b sea divisible por d, pues en caso contrario la congruencia (2.4) sería imposible para algún x entero. Por esta razón, suponiendo que b es un múltiplo de d, hacemos $a = a_1 d,\ b = b_1 d,\ m = m_1 d$. Entonces la congruencia (2.4) (después de haber simplificado por d) resulta equivalente a $a_1 x \equiv b_1 \mod m_1$, en la cual $(a_1,\ m_1) = 1$ y, por lo tanto admite una solución respecto del módulo m_1. Sea x_1 el resto no negativo mínimo de esta solución respecto del módulo m_1 entonces todos los números x que forman esta solución serán de la forma

$$x \equiv x_1 \mod m_1. \tag{2.5}$$

Respecto del módulo m los números (2.5) forman más de una solución; forman precisamente tantas soluciones cuantos números (2.5) haya en la sucesión $0,\ 1,\ 2, \ldots,\ m-1$ que sean resto no negativos mínimos respecto del módulo m.
Tales números son:

$$x_1,\ x_1 + m_1,\ x_1 + 2m_1, \ldots,\ x_1 + (d-1)m_1$$

es decir, en total d números satisfacen (2.5) y, por consiguiente, la congruencia (2.4) admite d soluciones.

Haciendo un resumen de todo lo demostrado, resulta el teorema siguiente:

Teorema 2.2.2 *Sea* $(a,\ m) = d$. *La congruencia* $ax \equiv b \mod m$ *es imposible si* b *no es divisible por* d. *Si* b *es múltiplo de* d, *la congruencia admite* d *soluciones.*

Para averiguar las soluciones de la congruencia (2.4), se indicará solamente un método, basado en la teoría de las fracciones continuas; además, es suficiente limitarse al caso $(a,\ m) = 1$.

Desarrollando en fracción continua la razón $m : a$

$$\frac{m}{a} = q_1 + \cfrac{1}{q_2 + \cfrac{1}{q_3 + \ddots + \cfrac{1}{q_n}}}$$

y considerando las dos fracciones reducidas últimas:

$$\frac{P_{n-1}}{Q_{n-1}},\ \frac{P_n}{Q_n} = \frac{m}{a}$$

en virtud de las propiedades de las fracciones continuas, se tiene:

$$mQ_{n-1} - aP_{n-1} = (-1)^n$$

$$aP_{n-1} \equiv (-1)^{n-1} \mod m$$
$$a(-1)^{n-1}P_{n-1}b \equiv b \mod m.$$

Así pues, la congruencia en cuestión admite la solución

$$x \equiv (-1)^{n-1}P_{n-1}b \mod m$$

para cuya solución es suficiente calcular P_{n-1}.

Ejemplo 2.2.3 Resolviendo la congruencia

$$111x \equiv 75 \mod 321 \tag{2.6}$$

Aquí $(111,\ 321) = 3$, siendo 75 múltiplo de 3. Por esta razón, la congruencia admite tres soluciones.
Dividiendo ambos miembros de la congruencia y el módulo por 3, obtenemos la congruencia

$$37x \equiv 25 \mod 107 \tag{2.7}$$

la cual debe resolverse primeramente. Se tiene

$$107 = 33 + (37)(2)$$

$$37 = 4 + (33)(1)$$

$$33 = 1 + (4)(8)$$

$$4 = 0 + (1)(4).$$

Por lo tanto, en este caso $n = 4,\ P_{n-1} = 26,\ b = 25$, y se obtiene la solución de la congruencia (2.7) en la forma

$$x \equiv -26 \cdot 25 \equiv 99 \mod 107.$$

De aquí, las soluciones de la congruencia (2.6) se expresan así:

$$x \equiv 99;\ 99 + 107;\ 99 + 2 \cdot 107 \mod 321$$

es decir

$$x \equiv 99;\ 206;\ 313 \mod 321.$$

2.2.3 Sistemas de congruencias de primer grado

En esta sección se estudiará el sistema más simple de congruencias con una incógnita, pero con distintos módulos que son primos dos a dos.

$$x \equiv b_1 \mod m_1,\ x \equiv b_2 \mod m_2,\ \ldots,\ x \equiv b_k \mod m_k. \tag{2.8}$$

Se puede resolver el sistema (2.8), es decir, se pueden hallar todos los valores de x que le satisfacen, aplicando el teorema siguiente:

Teorema 2.2.4 *Supongamos que los números M_s y M'_s vienen definidos por las condiciones*

$$m_1 m_2 \ldots m_k = M_s m_s,\ M_s M'_s \equiv 1 \mod m_s$$

y sea

$$x_0 = M_1 M'_1 b_1 + M_2 M'_2 b_2 + \ldots + M_k M'_k b_k.$$

Entonces el conjunto de valores de x que satisfacen al sistema (2.8) *se determina por la congruencia*

$$x_0 \equiv M_s M'_s b_s \equiv b_s \mod m_s. \tag{2.9}$$

Demostración En efecto, como todos los números M_j, distintos de M_s son divisibles por m_s, para cualquier $s = 1,\ 2\ , \ldots ,\ k$ se tiene

$$x_0 \equiv M_s M'_s b_s \equiv b_s \mod m_s$$

y por consiguiente, el sistema (2.8) es equivalente al sistema

$$x \equiv x_0 \mod m_1,\ x \equiv x_0 \mod m_2,\ \ldots,\ x \equiv x_0 \mod m_k \tag{2.10}$$

(es decir, a los sistemas (2.8) y (2.10) les satisfacen unos mismos valores de x). Pero, en virtud de los teoremas y proposiciones que se trataron en las primeras dos secciones, al sistema (2.10) le satisfacen aquellos valores de x, y sólo aquellos, que satisfacen a la congruencia (2.9). □

Proposición 2.2.5 *Si $b_1,\ b_2, \ldots,\ b_k$ recorren independientemente uno de otro los sistemas completos de restos respecto de los módulos $m_1,\ m_2, \ldots,\ m_k$, entonces x_0 recorre el sistema completo de restos respecto del módulo $m_1 m_2 \ldots m_k$.*

Demostración En efecto, x_0 recorre $m_1,\ m_2, \ldots,\ m_k$ valores, los cuales, debido a las proposiciones y teoremas de las dos primeras secciones son incongruentes respecto del módulo $m_1,\ m_2, \ldots,\ m_k$. □

Ejemplo 2.2.6 Resolviendo el sistema

$$x \equiv b_1 \mod 4,\ x \equiv b_2 \mod 5,\ x \equiv b_3 \mod 7$$

aquí $4 \cdot 5 \cdot 7 = 35 \cdot 4 = 28 \cdot 5 = 20 \cdot 7$, y además,

$$35 \cdot 3 \equiv 1 \mod 4,\ 28 \cdot 2 \equiv 1 \mod 5,\ 20 \cdot 6 \equiv 1 \mod 7$$

por lo tanto

$$x_0 = 35 \cdot 3b_1 + 28 \cdot 2b_2 + 20 \cdot 6b_3 = 105b_1 + 56b_2 + 120b_3$$

y por consiguiente, el conjunto de valores de x que satisfacen al sistema puede expresarse en la forma

$$x \equiv 105b_1 + 56b_2 + 120b_3 \mod 140.$$

Por ejemplo, el conjunto de valores de x que satisfacen al sistema

$$x \equiv 1 \mod 4,\ x \equiv 3 \mod 5,\ x \equiv 2 \mod 7$$

es

$$x \equiv 105 \cdot 1 + 56 \cdot 3 + 120 \cdot 2 \equiv 93 \mod 140$$

y el conjunto de valores de x que satisfacen al sistema

$$x \equiv 3 \mod 4,\ x \equiv 2 \mod 5,\ x \equiv 6 \mod 7,$$

es

$$x = 105 \cdot 3 + 56 \cdot 2 + 120 \cdot 6 \equiv 27 \mod 140.$$

2.2.4 Congruencias de cualquier grado respecto de un módulo primo

Sea p un número primo. Se demostrarán unos teoremas generales relativos a una congruencia de la forma

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

$$f(x) = a_0x^n + a_1x^{n-1} + \ldots + a_n. \tag{2.11}$$

Teorema 2.2.7 *Una congruencia de la forma* (2.11) *es equivalente a una congruencia de grado no superior a* $p-1$.

Demostración En efecto, dividiendo $f(x)$ por $x^p - x$, se tiene

$$f(x) = (x^p - x)Q(x) + R(x)$$

donde el grado de $R(x)$ no es superior a $p-1$. Como $x^p - x \equiv 0 \mod p$, resulta $f(x) \equiv R(x) \mod p$, de donde se deduce el teorema anterior. □

Teorema 2.2.8 *Si la congruencia* (2.11) *admite más de n soluciones, todos los coeficientes de* $f(x)$ *son múltiplos de p.*

Demostración Supóngase, que la congruencia (2.11) admite al menos $n+1$ soluciones. Designando los restos de estas soluciones con las letras $x_1,\ x_2, \ldots,\ x_n,\ x_{n+1}$ podemos expresar $f(x)$ en la forma

$$\begin{aligned} f(x) = \quad & a \quad (x-x_1)(x-x_2)\dots(x-x_{n-2})(x-x_{n-1})(x-x_n) + \\ & + \quad b(x-x_1)(x-x_2)\dots(x-x_{n-2})(x-x_{n-1}) + \\ & + \quad c(x-x_1)(x-x_2)\dots(x-x_{n-2}) + \\ & + \quad \cdots\cdots\cdots\cdots\cdots\cdots + \\ & + \quad k(x-x_1)(x-x_2) + \\ & + \quad l(x-x_1) \\ & + \quad m. \end{aligned} \tag{2.12}$$

De este modo, transformando los sumandos del segundo miembro en polinomios, elegimos b de tal modo que la suma de los coeficientes de x^{n-1} en los dos primeros polinomios coincida con a_1; una vez hallado b, elegimos c de tal modo que la suma de los coeficientes de x^{n-2} en los primeros tres polinomios coincida con a_2, etc.

Haciendo en (2.12) $x = x_1,\ x_2,\ \dots,\ x_n,\ x_{n+1}$, sucesivamente, se comprueba que todos los números $m,\ l,\ \dots,\ c,\ b,\ a$ son múltiplos de p. Por lo tanto, también son múltiplos de p todos los números $a,\ a_1,\ \dots,\ a_n$ (como sumas de números que son múltiplos de p). □

Teorema 2.2.9 *[Teorema de Wilson] Si p es un número primo, se verifica la congruencia*

$$1 \cdot 2 \dots \cdot (p-1) + 1 \equiv 0 \mod p. \tag{2.13}$$

Demostración En efecto, si $p = 2$ el teorema es evidente. Si $p > 2$ se considera la congruencia

$$(x-1)(x-2)\dots(x-(p-1)) - (x^{p-1}-1) \equiv 0 \mod p$$

ésta es de grado no superior a $p-2$ y admite $p-1$ soluciones; precisamente las soluciones cuyos restos son $1,\ 2,\ \dots,\ p-1$.

Por consiguiente, según el teorema anterior todos sus coeficientes son múltiplos de p; en particular, también es divisible por p el término independiente, el cual es precisamente igual al primer miembro de la congruencia (2.13). □

Ejemplo 2.2.10 Se tiene que $1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 \cdot 6 + 1 = 721 \equiv 0 \mod 7$.

2.2.5 Congruencias de cualquier grado respecto de un módulo compuesto

Teorema 2.2.11 *Si $m_1,\ m_2,\ \ldots,\ m_k$ son primos dos a dos, la congruencia*

$$f(x) \equiv 0 \mod (m_1 m_2 \ldots m_k) \tag{2.14}$$

es equivalente al sistema

$$f(x) \equiv 0 \mod m_1,\ f(x) \equiv 0 \mod m_2,\ \ldots,\ f(x) \equiv 0 \mod m_k.$$

Además, designando con $T_1,\ T_2,\ \ldots,\ T_k$ los números de soluciones de cada una de las congruencias de este sistema respecto de los módulos correspondientes y con T el número de soluciones de la congruencia (2.14)*, se tiene*

$$T = T_1 T_2 \ldots T_k.$$

Demostración En efecto, la primera parte del teorema se deduce de los incisos **c)** y **d)**, de la proposición 2.1.4. La segunda parte se deduce de que cada una de las congruencias

$$f(x) \equiv 0 \mod m_s \tag{2.15}$$

se cumple si y sólo si se cumple una de las T_s congruencias de la forma

$$x \equiv b_s \mod m_s$$

donde b_s recorre los restos de las soluciones de la congruencia (2.15); además, son posibles en total $T_1 T_2 \ldots T_k$ combinaciones distintas de la forma

$$x \equiv b_1 \mod m_1,\ x \equiv b_2 \mod m_2,\ \ldots,\ x \equiv b_k \mod m_k$$

que dan lugar a clases distintas respecto del módulo

$$m_1 m_2 \ldots m_k.$$

Ejemplo 2.2.12 La congruencia

$$f(x) \equiv 0 \mod 35,\ f(x) = x^4 + 2x^3 + 8x + 9 \tag{2.16}$$

es equivalente al sistema

$$f(x) \equiv 0 \mod 5,\ f(x) \equiv 0 \mod 7.$$

Fácilmente se comprueba que la primera congruencia de este sistema tiene 2 soluciones: $x = 1;\ 4 \mod 5$, la segunda congruencia tiene 3 soluciones: $x \equiv 3;\ 5;\ 6 \mod 7$.

Debido a esto, la congruencia (2.16) tiene $2 \cdot 3 = 6$ soluciones.
Para hallar estas 6 soluciones, hay que resolver 6 sistemas de la forma

$$x \equiv b_1 \mod 5,\ x \equiv b_2 \mod 7 \tag{2.17}$$

las cuales se obtienen haciendo recorrer a b_1 los valores $b_1 = 1;\ 4$, y a b_2 los valores $b_2 = 3;\ 5;\ 6$. Pero, como

$$35 = 7 \cdot 5 = 5 \cdot 7,\ 7 \cdot 3 = 1 \mod 5,\ 5 \cdot 3 = 1 \mod 7$$

el conjunto de valores de x que satisfacen al sistema (2.17) se expresa en la forma

$$x \equiv 21b_1 + 15b_2 \mod 35$$

por lo tanto, las soluciones de la congruencia (2.16) son:

$$x \equiv 31;\ 26;\ 6;\ 24;\ 19;\ 34 \mod 35.$$

Teorema 2.2.13 *La congruencia*

$$f(x) \equiv 0 \mod (p_1^{\alpha_1} p_2^{\alpha_2} \dots p_k^{\alpha_k})$$

es equivalente al sistema de congruencias de la forma

$$f(x) \equiv 0 \mod p^{\alpha} \tag{2.18}$$

la cual se reduce en general a la congruencia

$$f(x) \equiv 0 \mod p. \tag{2.19}$$

Demostración La primera equivalencia se deduce del teorema 2.2.11. Ahora cada x que satisface a la congruencia (2.18) necesariamente tiene que satisfacer también a la congruencia (2.19).

Sea

$$x \equiv x_1 \mod p$$

alguna solución de la congruencia (2.19). Entonces $x = x_1 + pt_1$, donde t_1 es entero. Poniendo este valor de x en la congruencia

$$f(x) \equiv 0 \mod p^2$$

y desarrollando el primer miembro según la fórmula de Taylor, se encuentra (teniendo en cuenta que $\frac{1}{k!}f^{(k)}(x_1)$ es entero y despreciando los términos que son múltiplos p^2):

$$f(x_1) + pt_1 f'(x_1) \equiv 0 \mod p^2, \quad \frac{f(x_1)}{p} + t_1 f'(x_1) \equiv 0 \mod p.$$

Limitándose aquí al caso en que $f'(x_1)$ no es divisible por p, resulta una solución:

$$t_1 \equiv t_1' \mod p; \; t_1 = t_1' + pt_2.$$

La expresión de x toma la forma

$$x = x_1 + pt_1' + p^2 t_2 = x_2 + p^2 t_2$$

sustituyendo en la congruencia

$$f(x) \equiv 0 \mod p^3$$

resulta que

$$f(x_2) + p^2 t_2 f'(x_2) \equiv 0 \mod p^3$$

$$\frac{f(x_2)}{p^2} + t_2 f'(x_2) \equiv 0 \mod p.$$

Aquí $f'(x_2)$ no es divisible por p, puesto que

$$x_2 \equiv x_1 \mod p;$$

$$f'(x_2) \equiv f'(x_1) \mod p$$

y por lo tanto, la última congruencia tiene una sola solución:

$$t_2 \equiv t_2' \mod p;$$

$$t_2 \equiv t_2' + pt_3.$$

La expresión de x toma la forma

$$x = x_2 + p^2 t_2' + p^3 t_3 = x_3 + p^3 t_3;$$

así sucesivamente. De este modo, partiendo de la solución dada de la congruencia (2.19) se tiene la solución congruente con ella de la congruencia (2.18). En resumen, toda solución $x \equiv x_1 \mod p$ de la congruencia (2.19), con la condición de que $f'(x_1)$ no sea divisible por p, proporciona una solución de la congruencia (2.18):

$$x = x_\alpha + p^\alpha t_\alpha$$

es decir

$$x \equiv x_\alpha \mod p^\alpha.$$

□

Ejemplo 2.2.14 Resolviendo la congruencia

$$\left.\begin{array}{l} f(x) \equiv 0 \mod 27 \\ f(x) = x^4 + 7x + 4 \end{array}\right\}. \tag{2.20}$$

La congruencia $f(x) \equiv 0 \mod 3$ tiene una solución $x \equiv 1 \mod 3$; en este caso $f'(1) \equiv 2 \mod 3$ y, por consiguiente, no es divisible por 3. Se tiene:

$$x = 1 + 3t_1$$
$$f(1) + 3t_1 f'(1) \equiv 0 \mod 9$$
$$3 + 3t_1 \cdot 2 \equiv 0 \mod 9$$
$$2t_1 + 1 \equiv 0 \mod 3$$
$$t_1 \equiv 1 \mod 3$$
$$t_1 = 1 + 3t_2$$
$$x = 4 + 9t_2$$
$$f(4) + 9t_2 f'(4) \equiv 0 \mod 27$$
$$18 + 9t_2 \cdot 2 \equiv 0 \mod 27$$
$$2t_2 + 2 \equiv 0 \mod 3$$
$$t_2 \equiv 2 \mod 3$$
$$t_2 = 2 + 3t_3$$
$$x = 22 + 27t_3.$$

Por lo tanto, la congruencia (2.20) tiene una solución

$$x \equiv 22 \mod 27.$$

Capítulo 3

Algoritmos de factorización

En este capítulo se presenta un algoritmo determinista de tiempo polinomial que determina si un número n de entrada es primo o compuesto. En la primera sección se describe el algoritmo, para realizar una prueba formal en la segunda sección en donde además se incluye el código en Maple para el algoritmo.
"El problema de distinguir números primos de números compuestos y de resolver el último en sus factores primos se sabe que es uno de los más importantes y útiles de la aritmética. Ha mantenido ocupados a la industria y la sabiduría de antiguos y modernos especialistas en geometría hasta tal punto que sería estupendo para discutir el problema largamente.... Además, la dignidad de la ciencia en sí misma parece requerir que cada uno de los posibles medios sean explorados para la solución de un problema a tal grado de ser elegante y célebre."

Karl Friedrich Gauss,*Disquisitiones Arithmeticae,*1801

3.1 Introducción

En esta sección se describe el algoritmo de Agrawal, Kayal y Saxena (AKS).

Uno de los problemas más estudiados en la historia de las matemáticas, ha sido el determinar si un número es primo o compuesto: desde la criba de Eratóstenes, pasando por Fermat, Lucas, y ya en nuestros tiempos, Solovay-Strassen y Miller-Rabin. En 1983, Adleman, Pomerance y Rumely presentan el primer algoritmo determinista, que posee una complejidad en tiempo $O(k \log n^{c \log\log\log n})$. Este algoritmo es conocido como el método ciclotómico

por usar precisamente los polinomios ciclotómicos, sin embargo, no es fácil de implementar. En 1986, Goldwasser y Killian proponen un algoritmo basado en curvas elípticas, corriendo a un tiempo polinomial. Sin embargo es poco eficiente en la práctica. Atkin propone otro algoritmo conocido como Elliptic Curve Primality Proving (ECPP) que es más eficiente en la práctica. Finalmente en 1992, Adleman y Huang modifican el algoritmo de Goldwasser y Killian para obtener un algoritmo probabilístico polinomial que siempre produce un número primo. Hasta agosto del 2002, el mejor algoritmo determinista era el de Adleman, Pomerance y Rumely con una complejidad superpolinomial y algunas modificaciones debidas a Mihailescu. El 6 de agosto del 2002, M. Agrawal, N. Kayal y N. Sexena encontraron un algoritmo polinomial determinista para certificar que un número es primo, cuando en efecto lo es. Esto quita del camino a la suposición de mostrar primero la conjetura de Riemann y después encontrar un algoritmo determinista de primalidad. El nuevo método, conocido como AKS, es simple y no requiere de grandes teorías y se ha considerado como uno de los grandes descubrimientos en esta área.

3.2 Algoritmo AKS-2002

El algoritmo AKS, en su primera versión de agosto del 2002, es determinista y posee una complejidad en tiempo $O((logn)^{12})$. Mas al igual que casi todo método, está sometido a un periodo de continuo perfeccionamiento. El algoritmo AKS-2002 se presenta a continuación:

`Entrada:` `entero` $n > 1$

1. `if(` n `es de la forma` a^b, $b > 1$ `) sale número COMPUESTO;`
2. $r = 2$;
3. `while`$(r < n)$ `{`
4. `if`$(\texttt{mcd}(n, r) \neq 1$ `) sale número COMPUESTO`
5. `if`$(r$ `es primo` `)`
6. `dar` q `factor primo suficientemente grande de` $r - 1$;
7. `if`$(q \geq 4\sqrt{r}\log n)$ `y` $(n^{\frac{r-1}{q}} \not\equiv 1 \mod r)$

8. `break;`
9. $r \leftarrow r + 1$;
10. `}`
11. `for` $a = 1$ `a` $2\sqrt{r}\log n$
12. `if`$((x-a)^n \not\equiv (x^n - a) \mod (x^r - 1, n))$ `sale número COMPUESTO;`
13. `sale número PRIMO;`

3.2.1 Idea del algoritmo

El siguiente teorema es una generalización del Teorema Pequeño de Fermat y puede tomarse como un criterio determinista.

Teorema 3.2.1 *Sea* $n \geq 2$ *y* $a < n$ *un entero tal que* $\text{mcd}(m, a) = 1$. *Entonces*

$$n \text{ es primo si y sólo si } (x + a)^n = (x^n + a) \mod n.$$

Sin embargo, para números de tamaño considerable, el criterio dado por este teorema no es eficiente, ya que requiere evaluar $O(n)$ coeficientes. Así, la idea básica de AKS fue reducir el número necesario de pasos para la verificación de la primalidad. La aportación de AKS es verificar sólo $(x + a)^n = (x^n + a) \mod (x^r - 1,\ n)$, para potencias r "pequeñas", donde $\mod (x^r - 1,\ n)$ significa evaluar los polinomios en el anillo cociente $\mathbb{Z}_n[x]/(x^r - 1)$. Resulta que todos los números primos cumplen este criterio. Sin embargo también algunos números compuestos lo cumplen para pocas parejas $(a,\ r)$, por lo que hay que descartar esos casos. El algoritmo elige primero (pasos 3-10) una potencia r adecuada, es decir, tal que sea un número primo y $r - 1$ tenga un factor q de valor alrededor de $r^{1/2}$. Se puede probar que tales potencias r existen. Finalmente el algoritmo verifica (pasos 11-13) que se cumpla la igualdad $(x + a)^n = (x^n + a) \mod (x^r - 1,\ n)$ en el anillo $\mathbb{Z}_n[x] = (x^r - 1)$, para $a = 1, \ldots,\ 2\sqrt{r}\log n$. En caso que así suceda, esto es suficiente para declarar que n es primo. Viendo primero paso a paso algunos detalles de implementación del algoritmo. Es importante mencionar que en aplicaciones criptográficas, donde es necesario generar un número primo de cierta longitud, se procede primero a generar un número aleatorio impar n, y éste se somete a alguna prueba de primalidad. Si se da como salida COMPUESTO, entonces

se intenta con $n+2$, y seguidamente con $n+4,\ n+6, \ldots$ hasta obtener el primo deseado.

A continuación se presenta una explicación más detallada de los cálculos que se realizan en cada paso:

Paso 1 Aquí se descarta que el número n sea una potencia b-ésima de un número entero a, es decir, se decide si acaso "n es una potencia perfecta". Para encontrar un posible exponente $b \geq 2$, se procede a calcular un entero $c = \lfloor n^{\frac{1}{b}} \rfloor$. Entonces resultará que n es una potencia perfecta si y sólo si $c^b = n$. Así pues, el paso 1 se puede verificar con la siguiente rutina:

```
for b = 2 a ⌊log n⌋
   if(c = n^(1/b)) es entero, sale número COMPUESTO
```

Paso 2 En esta etapa del algoritmo se busca un número primo r tal que $r-1$ tenga un factor $q \geq 4\sqrt{r}\log n$ tal que $q \mid o_r(n)$, donde $o_r(n)$ denota al orden de n módulo r. Este número r se busca de manera exhaustiva, es decir, se comienza desde $r = 2$ y se detiene hasta que se encuentra el deseado. La existencia del número r está garantizada por un resultado debido a E. Fouvry en 1985. El segundo propósito de esta etapa es detectar si acaso n tiene factores pequeños.

Paso 3 Aquí simplemente se inicializa $r = 2$. Realmente debe de inicializarse con $r = 3$ para verificar sólo los números impares. Se ha de proseguir en la búsqueda de r en tanto $r < n$.

Paso 4 Si $\text{mcd}(n,\ r) \neq 1$, entonces hay un divisor de n. Por lo tanto, el resultado del algoritmo ha de ser COMPUESTO.

Paso 5 Sólo se pasa al siguiente paso si r es primo, lo que se puede verificar mediante el ensayo de divisiones. Realmente se verifica únicamente a aquellos r impares; esto es, el incremento del paso 9 debe ser +2.

Paso 6 Aquí se localiza el factor primo q más grande de $r-1$. Mediante una búsqueda exhaustiva, se encuentra el factor impar más grande de $r-1$. Se puede realizar esta búsqueda mediante la siguiente rutina:

```
for i = 2 a r − 1
   while(r mod i = 0)
     r = r/i;
     se produce como factor i
```

Paso 7 Se verifica si acaso $q \geq 4\sqrt{r}\log n$ y si q divide al orden de n módulo r, mediante la prueba $n^{\frac{r-1}{q}} \not\equiv 1 \mod r$. Como q es primo y un factor de $r-1$, entonces si q no fuese un factor de $o_r(n)$ tendríamos que $\dfrac{(r-1)}{q} = o_r(n)k$. Por lo tanto $n^{\frac{r-1}{q}} \equiv 1 \mod r$. Es decir $o_r(n) \mid r-1$, pero $o_r(n) \mid \dfrac{(r-1)}{q}$, por lo cual $q \mid o_r(n)$.

Paso 8 Es importante notar que es necesario que n posea un tamaño al menos de 15 bits para poder encontrar el número r buscado.

Paso 9 Si r no es primo, se continúa buscando. La actualización debería de ser $r = r + 2$ que es el siguiente número impar.

Paso 10 Se termina la búsqueda de r. Esto queda garantizado por el resultado de Fouvry. En la última etapa del algoritmo, se revisa si acaso $(x-a)^n \equiv (x^n - a) \mod (n,\ x^r - 1)$ para $a = 1, \ldots,\ 2\sqrt{r}\log n$. Si no se cumpliera para algún a, entonces n ha de ser compuesto. De lo contrario n, será un número primo.

Paso 11 Se recorren los números a que verifican si acaso n satisface todas las congruencias. En tal caso, n es primo.

Paso 12 Si se encuentra a tal que $(x-a)^n \not\equiv (x^n - a) \mod (x^r - 1,\ n)$ entonces n es compuesto.

Paso 13 Si, finalmente, n cumple todas las equivalencias del paso 12, con el r encontrado, entonces n es primo.

3.2.2 Una versión nueva del algoritmo AKS

Con observaciones de H. Lenstra, se publica por los mismos autores una nueva versión del algoritmo AKS, la cual se muestra a continuación:

Entrada: entero $n > 1$

1. if(n es de la forma a^b, $b > 1$) sale número COMPUESTO;
2. encontrar el r mas pequeño tal que $o_r(n) > 4\log^2 n$;
3. if$(1 < \gcd(a,\ n) < n)$, para algún $a \leq r$, sale número COMPUESTO;
4. if$(n \leq r)$ sale número PRIMO;
5. for $a = 1$ a $\lfloor 2\sqrt{\varphi(r)}\log n \rfloor$
   if$((x+a)^n \not\equiv (x^n + a) \mod (x^r - 1,\ n))$, sale número COMPUESTO;
6. sale número PRIMO;

Esta nueva versión del algoritmo tiene también tres etapas, la primera revisa si n es una potencia perfecta. Los métodos aquí conocidos pueden perfeccionarse hasta lograr un tiempo lineal [5, 6]. La segunda etapa consiste de nuevo en buscar un r adecuado y también eliminar la posibilidad de que n tenga factores pequeños. Finalmente la última etapa consiste en probar para $a = 1$ hasta $\lfloor 2\sqrt{\varphi(r)}\log n \rfloor$ que se cumpla $(x+a)^n \equiv (x^n + a) \mod (x^r - 1,\ n)$. En tal caso, se declara que n es primo. Una vez más el esfuerzo mayor se hace en evaluar $(x+a)^n \mod (x^r - 1)$ y particularmente, como se observa en el paso 5 del último algoritmo, si n fuese primo es necesario evaluar todos las a's posibles.

3.2.3 Idea de la demostración del algoritmo AKS

De manera más formal, el resultado de AKS fue escrito por D. Bernstein, y se resume en el siguiente teorema.

Teorema 3.2.2 *[Agrawal, Kayal, Saxena] Sea n un entero positivo, sean q, r dos números primos y sea S un conjunto finito de enteros. Suponiendo que:*

1. *q divide a r.*
2. $n^{\frac{(r-1)}{q}} \mod r,\ \notin \{0,\ 1\}$.

3. $\text{mcd}(n,\ b - b') = 1$ *para todos los elementos diferentes* $b,\ b' \in S$.

4. $\binom{|S| + q - 1}{|S|} \geq n^{2\lfloor\sqrt{r}\rfloor}$.

5. $(x + b)^n = x^n + b$ *en el anillo* $\mathbb{Z}_n[x]/(x^r - 1)$, *para todo* $b \in S$, *entonces* n *es una potencia de un primo.*

Posteriormente Lenstra propone refinar el resultado reformulándolo de la siguiente manera:

Teorema 3.2.3 *[Agrawal, Kayal, Saxena, Lenstra] Sean* n, r *dos enteros positivos y sea* S *un conjunto finito de enteros. Suponiendo que:*

1. n *es una raíz primitiva módulo* r.

2. $\text{mcd}(n,\ b - b') = 1$ *para todos los elementos diferentes* $b,\ b' \in S$.

3. $\binom{|S| + q - 1}{|S|} \geq n^{2\lfloor\sqrt{r}\rfloor}$.

4. $(x + b)^n = x^n + b$ *en el anillo* $\mathbb{Z}_n[x]/(x^r - 1)$, *para todo* $b \in S$, *entonces* n *es una potencia de un primo.*

Para mostrar que el algoritmo es correcto hay que verificar dos cosas: primero, si n es primo entonces el algoritmo debe dar como salida PRIMO, pero esto es claro, ya que n no es una potencia perfecta ni posee factores pequeños ni satisface el teorema de AKS. La parte difícil es demostrar que si el algoritmo produce como salida PRIMO, entonces en efecto n es primo. Para esto último, se construye de manera conveniente un grupo cíclico G a partir de los elementos r, q dados por el algoritmo. G es un subgrupo del grupo multiplicativo $(F_p[x]/h(x))^*$, siendo $h(x)$ un polinomio irreducible. Entonces se ha de mostrar que necesariamente n es una potencia del primo p o bien n tiene factores pequeños. Mas como esta segunda posibilidad ha sido descartada previamente, se ha de tener que n es primo. Para ver un bosquejo de la demostración del teorema AKS se puede consultar [12]. Para la demostración

completa, detallada y presentada de varias maneras diferentes se puede consultar [1, 2, 3, 4, 11, 13, 16].

En la siguiente sección se hace un análisis de la demostracin formal del resultado.

3.3 Demostración formal del algoritmo AKS

3.3.1 Historia

Desde épocas antiguas, problemas referentes a números primos han fascinado a los matemáticos. Uno de los problemas fundamentales referentes a números primos está en determinar si un número dado es primo. En tiempos modernos, la prueba de primalidad también ha llegado a ser importante desde una perspectiva práctica debido a sus aplicaciones en criptografía.

Partiendo desde los chinos y griegos antiguos, muchos han trabajado en el problema de buscar un algoritmo de prueba de primalidad. Como ya se mencionó anteriormente, la criba de Eratóstenes (ca. 240 A.C.) es el algoritmo más antiguo en el que se basan los trabajos para detectar si un número es primo, sin embargo, su complejidad del tiempo ($= \Omega(n)$ donde n es el número de entrada) es exponencial. En el siglo XVII, Fermat probó en el Teorema pequeño de Fermat que para cualquier número primo p, y cualquier número a no divisible por p, $a^{p-1} = 1(\mod p)$.

Aunque la doble implicación de este teorema no se cumple, este resultado ha sido el punto de partida para la prueba de primalidad de varios algoritmos. En 1976, Miller [34] utilizó esta propiedad para obtener un algoritmo determinista de tiempo polinomial para obtener la prueba de Hipótesis Extendida de Riemann (Extended Riemann Hypothesis o ERH). Su prueba fue modificada por Rabin [35] para producir un algoritmo de tiempo polinomial sin condiciones pero aleatorio. Solovay y Strassen [36] obtuvieron otro algoritmo de tiempo polinomial aleatorio y usando residuos cuadráticos. (Su algoritmo puede también estar desaleatorizado bajo la suposición de ERH). Desde entonces, una gran cantidad de algoritmos aleatorios de tiempo polinomial se han propuesto para la prueba de primalidad.

En 1983, Adleman, Pomerance, y Rumely alcanzaron una brecha importante dando un algoritmo determinista para la prueba de primalidad que es del orden $(\log n)^{O(\log\log\log n)}$ (todos los algoritmos deterministas anteriores al de Adleman, Pomerance, y Rumely requieren un tiempo exponencial). En 1986,

Goldwasser y Kilian [29] propusieron un algoritmo aleatorio basado en curvas elípticas que funcionaba en tiempo polinomial en casi todas las entradas que produce un certificado para la prueba de primalidad (hasta entonces, todos los certificados producidos por algoritmos aleatorios eran solo para numeros compuestos). Un algoritmo similar fue desarrollado por Atkin [24].

Adleman y Huang [21] modificaron el algoritmo de Goldwasser-Kilian para obtener un algoritmo de tiempo polinomial de selección aleatoria que produce siempre una prueba para verificar si un número es o no primo.

La última meta de esta línea de investigación es, por supuesto, obtener un algoritmo determinista incondicional de tiempo polinomial para probar si un número es o no primo. A pesar del progreso impresionante hecho al comprobar si un número es o no primo que se ha mencionado hasta ahora, esta meta no se ha podido alcanzar. Sin embargo, al algoritmo AKS es un modelo determinista, de tiempo polinomial de orden $\widetilde{O}((\log n)^{12})$ en el cual prueba si un número es o no primo.

3.3.2 Enfoque e idea básica

Éste algoritmo (AKS) está basado en el seguimiento de la identidad de números primos. Esta misma identidad fue básica para el algoritmo aleatorio de tiempo polinomial en [20]:

Proposición 3.3.1 *[Identidad] Suponga que a es coprimo de p. Entonces p es primo si y sólo si*

$$(x-a)^p \equiv (x^p - a) \mod p. \tag{3.1}$$

Demostración Para $0 < i < p$, el coeficiente de x^i en $((x-a)^p - (x^p - a))$ es $(-1)^i \binom{p}{i} a^{p-i}$. Ahora si p es primo, $\binom{p}{i} \equiv 0 \mod p$ y por lo tanto todos los coeficientes son cero. Si p es compuesto: considerar un número primo q que es factor de p y se tiene que $q^k \parallel p$. Luego q^k no es divisor de $\binom{p}{q}$ y es coprimo de a^{p-q} y por lo tanto el coeficiente de x^q no es $0 \mod p$. Así $((x-a)^p - (x^p - a))$ no es idénticamente cero sobre F_p. □

Así dando p de entrada, uno puede escoger el polinomio $P(x) = x - a$ y calcular si la congruencia (3.1) se satisface o no. Sin embargo,en el peor de los casos, esto lleva un tiempo $\Omega(p)$ porque es necesario evaluar el coeficiente de grado p. Por lo tanto, para que sea factible se evaluan ambos lados en (3.1) tanto

en el módulo como en el polinomio de la forma $x^r - 1$. Una iteración en éste algoritmo consiste en la evaluación de los siguientes contenidos:

$$(x-a)^p \equiv (x^p - a) \mod (x^r - 1, p) \tag{3.2}$$

De la proposición 3.3.1 [Identidad] esto es inmediato para todos los números primos p se satisface la congruencia anterior para todos los valores de a y r; sin embargo, algunos números compuestos p pueden satisfacer (3.2) para algunos valores de (a, r). La congruencia anterior toma un tiempo de verificación de $O(r^2 \log^3 p)$, o aún mejor $O(r \log^2 p)$ si se usa la Multiplicación Rápida de Fourier [31]. Este algoritmo primero escoge un r "apropiado". (Y r es "apropiado" si es primo $= O(\log^6 p)$ y contiene $r-1$ factores primos y con un tamaño de por lo menos $r^{\frac{1}{2}+\delta}$, para algunas constantes $\delta > 0$. [27,25] nos asegura que tal r "apropiado" existe.). Posteriormente, el algoritmo verifica la congruencia (3.2) para un número "pequeño" $(O(\sqrt{r} \log p))$ de a's. Posteriormente se probará si esta idea funciona; es decir, el algoritmo correcto que determina si p es un número primo o no.

3.3.3 Notación y preliminares

Esta sección declara algunos resultados de teoría algebráica y teoría de números los cuales se van a usar en demostraciones posteriores. En el resto del texto F_{p^d} denota un campo finito donde p es primo. Cabe recordar eso, si p es primo y $h(x)$ es un polinomio de grado d e irreducible en F_p, luego $F_p[x]/(h(x))$ es un campo finito de orden p^d. En el resto del texto $h(x)$ es un factor de $\dfrac{x^r - 1}{x - 1}$ a menos que se indique de otra manera.

Se usará el símbolo $\widetilde{O}(t(n))$ para $O(t(n)\text{poly}(\log t(n)))$, donde $t(n)$ es cierta función que depende de n. A menos que se indique lo contrario el logaritmo es de base 2 en lo subsiguiente.

Ahora se mostrarán algunos hechos simples de álgebra que se pueden encontrar en cualquier texto especializado en ésta materia.

Lema 3.3.2 *Dados p y r números primos $p \neq r$.*

1. *El grupo multiplicativo de cualquier campo F_{p^t} para $t > 0$, se denota por $F^*_{p^t}$ que es cíclico.*

2. *Dado $f(x)$ un polinomio con coeficientes enteros. Se tiene que*

$$f(x)^p \equiv f(x^p) \mod p.$$

3. *Dado $h(x)$ cualquier factor de $x^r - 1$. Dando $m \equiv m_r \mod r$. Se tiene que*

$$x^m \equiv x^{m_r} \mod h(x).$$

4. *Dado $o_r(p)$ el orden p módulo r. Luego en F_p, descomponiendo los polinomios $\dfrac{x^r-1}{x-1}$ en factores irreducibles cada uno de grado $o_r(p)$.*

Demostración

1. Ver ejemplo [33].

2. Dado $f(x) = a_0 + a_1 x + \cdots + a_d x^d$. El coeficiente de x^i en $f(x)^p$ es

$$\sum_{\substack{i_0+\cdots+i_d=p \\ i_1+2i_2+\cdots+di_d=i}} a_0^{i_0} \cdots a_d^{i_d} \frac{p!}{i_0! \cdots i_d!}.$$

   Note que esta sumatoria es divisible por p a menos que uno de los i_j's sea p. En el caso siguiente $i = pj$ y el coeficiente de x^i es $a_j^p = a_j$. Estas son las congruencias que requerimos.

3. Dado $m = kr + m_r$. Ahora

$$\begin{array}{lrcll} & x^r & \equiv & 1 & \mod x^r - 1 \\ \text{implica} & x^{kr} & \equiv & 1 & \mod x^r - 1 \\ \text{implica} & x^{kr+m_r} & \equiv & x^{m_r} & \mod x^r - 1 \\ \text{implica} & x^m & \equiv & x^{m_r} & \mod h(x). \end{array}$$

4. Dado $d = o_r(p)$ y $Q_r(x) = \dfrac{x^r-1}{x-1}$. Supongase que $Q_r(x)$ tiene un factor irreducible, $h(x)$ en F_p de grado k. Ahora formamos $F_p[x]/h(x)$ un campo de tamaño p^k y un subgrupo multiplicativo de $F_p[x]/h(x)$ que es cíclico con un generador llamado $g(x)$. Asimismo, en el campo de Galois, de la ecuación mencionada en el lema 3.3.2 punto *2*. se tiene:

$$\begin{array}{lrcll} & g(x)^p & \equiv & g(x^p) & \\ \text{implica} & g(x)^{p^d} & \equiv & g(x^{p^d}) & \\ \text{implica} & g(x)^{p^d} & \equiv & g(x) & \text{[Lema 3.3.2 punto } 3.\text{]} \\ \text{implica} & g(x)^{p^d-1} & \equiv & 1. & \end{array}$$

Puesto que $(p^k - 1)$ es del orden $g(x)$, se tiene $(p^k - 1)|(p^d - 1)$ esto implica que $k|d$.
También se tiene que $h(x)|(x^r - 1)$ en F_p y posteriormente en el campo $F_p[x]/h(x)$ se tiene

$$x^r \equiv 1.$$

Así el orden de x en este campo tiene que ser r (puesto que r es primo y $x \not\equiv 1$). Posteriormente $r|(p^k - 1)$, es decir, $p^k \equiv 1 \mod r$. Por lo tanto, $d|k$. Luego $k = d$, por lo que el lema se cumple. □

Además de los hechos algebráicos anteriores, se necesitan los dos hechos teóricos numéricos siguientes.

Lema 3.3.3 [27, 25] *Dado $P(n)$ que denota el divisor primo más grande de n. Existen constantes $c > 0$ y n tal que, para todo $x \geq n_0$*

$$|\{p \backslash p \text{ es primo, } p \leq x \text{ y } P(p-1) > x^{\frac{2}{3}}\}| \geq c\frac{x}{\log x}.$$

Lema 3.3.4 [22] *Dado $\pi(n)$, que denota el número de primos que son menores o iguales a n. Se tiene que para $n \geq 1$:*

$$\frac{n}{6\log n} \leq \pi(n) \leq \frac{8n}{\log n}.$$

3.3.4 El Algoritmo

Teorema 3.3.5 *El algoritmo AKS regresa PRIMO si y sólo si n es un número primo.*

En el resto de esta sección, se va a establecer este teorema a través de la sucesión de lemas. Primero note que el algoritmo tiene dos nudos. El primer nudo intenta encontrar un número primo r tal que $r - 1$ tiene un factor primo grande $q \geq 4\sqrt{r}\log n$, y que $q|o_r(n)$, donde $o_r(n)$ es del orden de n módulo r. Por lo que se limitará el número de iteraciones del nudo while después de que se encuentra tal r.

Lema 3.3.6 *Existen constantes positivas c_1 y c_2 para las cuales r es primo en el intervalo $[c_1(\log n)^6, c_2(\log n)^6]$ tal que $r - 1$ tiene un factor primo $q \geq 4\sqrt{r}\log n$ y $q|o_r(n)$.*

Demostración Dados c y $P(n)$ según el lema (3.3.3). Así, el número primo de r's (llamados primos especiales) entre $c_1(\log n)^6$ y $c_2(\log n)^6$ tal que $P(r-1) > (c_2(\log n)^6)^{\frac{2}{3}} > r^{\frac{2}{3}}$ es (para n suficientemente grande)

$$\begin{aligned} &\geq [1 \cdots c_2(\log n)^6] - [1 \cdots c_1(\log n)^6] \\ &\geq \frac{cc_2(\log n)^6}{7 \log\log n} - \frac{8c_1(\log n)^6}{6 \log\log n} \text{(Usando el lema 3.3.4)} \\ &= \frac{(\log n)^6}{\log\log n}\left(\frac{cc_2}{7} - \frac{8c_1}{6}\right). \end{aligned}$$

Eligiendo constantes $c_1 \geq 4^6$ y c_2 de modo que la cantidad en pares sea una constante positiva, llamada c_3.

Dado $x = c_2(\log n)^6$. Considere el producto

$$\prod = (n-1)(n^2-1)\cdots(n^{x^{\frac{1}{3}}}-1).$$

Este producto tiene a lo más $x^{\frac{2}{3}} \log n$ factores primos.
Note que:

$$x^{\frac{2}{3}} \log n < \frac{c_3(\log n)^6}{\log\log n}.$$

Posteriormente hay un número primo especial, llamado r, que no divide al producto $\prod$.

Este es el número primo requerido: $r-1$ que tiene un factor primo grande, esto es, $q \geq r^{\frac{2}{3}} \geq 4\sqrt{r}\log n$ (puesto que $c_1 \geq 4^6$), y $q | o_r(n)$. □

Una vez que el nudo while se detiene, se puede mostrar el teorema 3.3.5:

Demostración El nudo while no regresa COMPUESTO ya que el $\text{mcd}(n, r) = 1$ para todo $r \leq c_2(\log n)^6$, donde c_2 está definido como en el lema 3.3.6. Con el lema 3.3.2 (en el punto *2.*), el nudo for tampoco regresa COMPUESTO. Así, el algoritmo identificará a n como PRIMO. □

Ahora es el turno de poner atención en el caso en donde de entrada n es compuesto en el algoritmo. La importancia de establecer a r en el nudo while surge cuando n es compuesto y llamamos p_i, $1 \leq i \leq k$, como sus factores primos.

En este caso $o_r(n)|\text{mcm}_i\{o_r(p_i)\}$ y por lo tanto existe un factor primo p de n tal que $q|o_r(p)$, donde q es el factor primo mas grande de $r-1$.

El segundo nudo del algoritmo usa el valor obtenido de r para hacer operaciones polinómicas a $l = 2\sqrt{r}\log n$ binomiales: $x-a$ para $1 \leq a \leq l$. Con el lema 3.3.2 en el punto *4.*, se tiene el polinomio $h(x)$ (factor de x^r-1) de grado $d = o_r(p)$ irreducible en F_p. Note que

$$(x-a)^n \equiv (x^n - a) \mod (x^r - 1,\ n)$$

implica que

$$(x-a)^n \equiv (x^n - a) \mod (h(x),\ p).$$

Son las identidades que se tienen en cada binomio en el campo $F_p[x]/(h(x))$. El conjunto de l formas binomiales en un grupo cíclico grande en éste campo:

Lema 3.3.7 *En el campo $F_p[x]/(h(x))$, el grupo generado por las l binomiales: $(x-a)$, $1 \leq a \leq l$, es decir,*

$$G = \left\{ \prod_{1\leq a\leq l} (x-a)^{\alpha_a} | \alpha_a \geq 0,\ \textit{para todo } 1 \leq a \leq l \right\}$$

es cíclico y de tamaño mayor que $\left(\frac{d}{l}\right)^l$.

Demostración Esta claro que G es un grupo y puesto que es un subgrupo del grupo cíclico $(F_p[x]/(h(x)))^*$, es también cíclico.
Ahora considerando el conjunto

$$S = \left\{ \prod_{1\leq a\leq l} (x-a)^{\alpha_a} | \sum_{1\leq a\leq l} \alpha_a \leq d-1,\ \alpha_a \geq 0,\ \text{para todo } 1 \leq a \leq l \right\}.$$

El argumento se sigue al demostrar que todos los elementos de S son distintos en $F_p[x]/(h(x))$. El nudo while asegura esto una vez que pare el último r tal que $r > q > 4\sqrt{r}\log n > l$. También el paso 4 del algoritmo comprueba que el mcd de r y n es 1 si cualquiera de las a's son congruentes con el módulo p.
Así que cualesquiera dos elementos de S son distintos módulo p. Esto implica

que todos los elementos de S son distintos en el campo $F_p[x]/(h(x))$ puesto que el grado de cualquier elemento de S es menor que $d - el\ grado\ de\ h(x)$. La cardinalidad del conjunto S es:

$$\begin{aligned}\binom{l+d-1}{l} &= \frac{(l+d-1)(l+d-2)\cdots(d)}{l!} \\ &> \left(\frac{d}{l}\right)^l.\end{aligned}$$

Puesto que S es justamente un subconjunto de G se tiene el resultado utilizando el teorema 1.3.34. □

Dado que $d \geq 2l$, la dimensión de G es mayor que $2^l = n^{2\sqrt{r}}$. Definiendo ahora un conjunto relacionado con $g(x)$ el cual desempeñará un papel importante en las discusiones restantes.

$$I_{g(x)} =: \left\{m | g(x)^m \equiv g(x^m) \mod (x^r - 1, p)\right\}.$$

Se tiene la siguiente propiedad de $I_g(x)$:

Lema 3.3.8 *El conjunto $I_g(x)$ es cerrado bajo la multiplicación.*

Demostración Dados m_1,$m_2 \in I_g(x)$. Se tiene

$$g(x)^{m_1} \equiv g(x^{m_1}) \mod (x^r - 1,\ p)$$

y

$$g(x)^{m_2} \equiv g(x^{m_2}) \mod (x^r - 1,\ p).$$

También se tiene que al sustituir x^{m_1} en lugar de x en la segunda congruencia:

$$\begin{aligned} g(x^{m_1})^{m_2} &\equiv g(x^{m_1 m_2}) \mod (x^{m_1 r} - 1,\ p) \\ \text{implica}\quad g(x^{m_1})^{m_2} &\equiv g(x^{m_1 m_2}) \mod (x^r - 1,\ p). \end{aligned}$$

De lo anterior se tiene que

$$\begin{aligned} g(x)^{m_1 m_2} &\equiv (g(x)^{m_1})^{m_2} \mod (x^r - 1,\ p) \\ &\equiv g(x^{m_1})^{m_2} \mod (x^r - 1,\ p) \\ &\equiv g(x^{m_1 m_2}) \mod (x^r - 1,\ p). \end{aligned}$$

Por lo tanto m_1,$m_2 \in I_g(x)$. □

Ahora se probará una propiedad de $I_g(x)$ que desempeña un papel crucial en la demostración.

Lema 3.3.9 *Sea o el orden de $g(x)$ en $F_p[x]/(h(x))$ y m_1, $m_2 \in I_g(x)$. Si $m_1 \equiv m_2 \mod r$ entonces $m_1 \equiv m_2 \mod o_g$.*

Demostración Puesto que $m_1 \equiv m_2 \mod r$, $m_2 = m_1 + kr$ para algun $k \geq 0$. Y dado que $m_2 \in I_g(x)$: (en lo siguiente las congruencias están en el campo $F_p[x]/(h(x))$ a menos que se indique lo contrario)

$$\begin{array}{lrcl} & g(x)^{m_2} & \equiv & g(x)^{m_2} \mod (x^r - 1,\ p) \\ \text{implica} & g(x)^{m_2} & \equiv & g(x^{m_2}) \\ \text{implica} & g(x)^{m_1+kr} & \equiv & g(x^{m_1+kr}) \\ \text{implica} & g(x)^{m_1} g(x)^{kr} & \equiv & g(x^{m_1}) \quad \text{[Por el lema 3.3.2 punto } 3\text{.]} \\ \text{implica} & g(x)^{m_1} g(x)^{kr} & \equiv & g(x)^{m_1}. \end{array}$$

Ahora $g(x) \not\equiv 0$ implica que $g(x)^{m_1} \not\equiv 0$ y por lo tanto se puede eliminar $g(x)^{m_1}$ de ambos lados por lo que

$$g(x)^{kr} \equiv 1.$$

Posteriormente,

$$\begin{array}{rcl} kr & \equiv & 0 \mod o_g \\ \text{implica} \quad m_2 & \equiv & m_1 \mod o_g \end{array}$$

por lo que el lema se cumple. □

La propiedad anterior implica que hay "muy pocos" números ($\leq r$) en $I_{g(x)}$ que son menores que o_g.
Ahora es posible demostrar la propiedad más importante del algoritmo.

Lema 3.3.10 *Si n es compuesto el algoritmo regresa COMPUESTO.*

Demostración Supongase que el algoritmo regresa PRIMO (en lugar de COMPUESTO). Así, el nudo del for asegura que para todos $1 \leq a \leq 2\sqrt{r}\log n$,

$$(x - a)^n \equiv (x^n - a) \mod (x^r - 1,\ p). \tag{3.3}$$

Note que $g(x)$ es justo un producto de potencias de l binomiales $(x-a)$, $(1 \leq a \leq l)$ las cuales satisfacen la ecuación (3.3). Así,

$$g(x)^n \equiv g(x^n) \mod (x^r - 1,\ p).$$

Tomando $n,\ p \in I_{g(x)}$, por el lema 3.3.2 en el punto *2.*, y trivialmente, $1 \in I_{g(x)}$. Se quiere ahora demostrar que el conjunto $I_{g(x)}$ tiene "muchos" números que son menores que o_g lo cual contradice al lema 3.3.9.
Considere el conjunto

$$E = \left\{ n^i p^i |\ 0 \leq i, j \leq \lfloor \sqrt{r} \rfloor \right\}$$

con el lema 3.3.8 $E \subseteq I_{g(x)}$. Puesto que $|E| = (1 + \lfloor \sqrt{r} \rfloor)^2 > r$, hay dos elementos $n^{i_1}p^{j_1}$ y $n^{i_2}p^{j_2}$ en E con $i_1 \neq i_2$ o $j_1 \neq j_2$ tal que $n^{i_1}p^{j_1} \equiv n^{i_2}p^{j_2}(\mod r)$ por el principio de clasificación. Pero por el lema 3.3.9 se tiene $n^{i_1}p^{j_1} \equiv n^{i_2}p^{j_2} \mod o_g$. Ésto implica que

$$n^{i_1 - i_2} \equiv p^{j_2 - j_1}(\mod o_g).$$

Puesto que $o_g > n^{2\sqrt{r}}$ y $n^{|i_1 - i_2|},\ p^{|j_1 - j_2|} < n^{\sqrt{r}}$ en la congruencia anterior se tiene una igualdad. Dado que p es primo, esta igualdad implica que $n = p^k$ para algún $k \geq 1$. Sin embargo, en el paso 1 del algoritmo, los números compuestos de la forma p^k para $k \geq 2$ ya son detectados. Luego se tiene que, $n = p$ lo cual se contradice. □

3.3.5 Análisis de la complejidad del tiempo

Es posible calcular de forma directa la complejidad del tiempo del algoritmo.

Teorema 3.3.11 *La complejidad del tiempo asintótico del algoritmo es*

$$\widetilde{O}(\log^{12} n).$$

Demostración El primer paso del algoritmo toma un tiempo asintótico: $O(\log^3 n)$. Según lo observado durante el análisis del algoritmo en la sección anterior, el nudo del while hace $O(\log^6 n)$ iteraciones.

Ahora midamos el trabajo hecho en una iteración del nudo while. En el primer paso (cálculo del mcd) tomamos un tiempo asintótico poly$(\log \log r)$. Los dos

pasos siguientes tomarían a lo mas un tiempo de $r^{\frac{1}{2}}\text{poly}(\log\log n)$ en la implementación de la fuerza bruta.

Los tres siguientes pasos tomarían a lo más $\text{poly}(\log\log n)$ pasos. Así el tiempo asintótico total tomandose desde el nudo while es $\widetilde{O}(\log^6 n \cdot r^{\frac{1}{2}}) = \widetilde{O}(\log^9 n)$.

El nudo del for hace el cálculo modular sobre los polinomios. Si la Multiplicación Rápida de Fourier es usada entonces una iteración en este nudo toma $\widetilde{O}(\log n \cdot r \log n)$ pasos. Así, el nudo del for toma un tiempo asintótico $\widetilde{O}(r^{\frac{3}{2}} \log^3 n) = \widetilde{O}(\log^{12} n)$. □

En la práctica, sin embargo, este algoritmo puede trabajar mucho más rápidamente. La razón es que aún cuando solamente se sabe que hay "muchos" números primos r tal que $P(r-1) > r^{\frac{2}{3}}$, una propiedad más fuerte es creer que es verdad. En realidad esto es verdadero para muchos números primos r, $P(r-1) = \dfrac{r-1}{2}$ (tales números primos son llamados números primos de Sophie Germain).

Definición 3.3.12 *Si ambos r y $\dfrac{r-1}{2}$ son primos, entonces $\dfrac{r-1}{2}$ es un número primo de Sophie Germain. Llamaremos a los r's como números coprimos de Sophie Germain.*

La siguiente conjetura es de la densidad de los números primos de Sophie Germain. Esta conjetura se ha verificado para $r \leq 10^{10}$:

Conjetura 3.3.13 [30] *El número coprimo de Sophie Germain es asintótico a $\dfrac{D_x}{\log^2 x}$, donde D es el mismo número primo constante (estimado con Wrench y otros que son aproximadamente* $0.6601618\ldots$*).*

Si esta conjetura es cierta, entonces el nudo while existe para un r "apropiado" del tamaño $\widetilde{O}(\log^2 n)$:

Lema 3.3.14 *Asumiendo la conjetura, existe r "apropiado" en el rango de $64\log^2 n$ a $c_2 \log^2 n$ para todo $n > n_0$, donde n_0 y c_2 son constantes positivas.*

Demostración Primero en todo el texto si r es un número primo y $q = \dfrac{r-1}{2}$ es un número primo, entonces el orden único de n módulo r son $\{1, 2, q, 2q = r-1\}$. Pero el orden de n módulo r puede ser 1 o 2 para a lo mas $2\log n$ números primos r; (esto es porque (n^2-1) puede tener a lo mas $\log(n^2-1)$ factores primos). Dejando a un lado estos factores primos de (n^2-1) y considerando el orden de los números coprimos r de Sophie Germain para el cual el orden de n módulo r es por lo menos $\dfrac{r-1}{2}$. Ahora con

$$\begin{array}{rrcl} & \frac{r-1}{2} & \geq & 4\sqrt{r}\log n \\ \text{implica} & \sqrt{r} & \geq & 8\log n \\ \text{implica} & r & \geq & 64\log^2 n. \end{array}$$

Por lo tanto considerando el rango de $64\log^2 n$ a $c_2\log^2 n$ y demostrando que eligiendo a c_2 bastante grande, se encuentra por lo menos un r deseado en este rango. Con la conjetura 3.3.13, hay $\dfrac{D_{c_2}\log^2 n}{\log^2(c_2\log^2 n)}$ números coprimos de Sophie Germain menos que $c_2\log^2 n$. Fuera de ésto, a lo más $\dfrac{D64\log^2 n}{\log^2(64\log^2 n)}$ son menores que $64\log^2 n$ (otra vez debido a la conjetura 3.3.13).

De los restantes hay a lo mas $2\log n$ números primos para que el orden de n módulo r sea 1 o 2. Así elegimos a c_2 tal que:

$$\begin{array}{rrcl} & \dfrac{D_{c_2}\log^2 n}{\log^2(c_2\log^2 n)} & > & \dfrac{D64\log^2 n}{\log^2(64\log^2 n)} + 2\log n \\ o, & \dfrac{c_2\log^2 n}{(\log\log n)^2} & > & \dfrac{100\log^2 n}{(\log\log n)^2} \\ o, & c_2 & > & 100 \end{array}$$

por lo que tal r existe. □

Ésto nos conduce inmediatamente a una complejidad heurística del tiempo $\widetilde{O}(r^{\frac{1}{2}}(\log n)^2)$ (del nudo while) + $\widetilde{O}(r^{\frac{3}{2}}(\log n)^3)$ (del nudo for) = $\widetilde{O}(\log^6 n)$ para este algoritmo.

3.3.6 Trabajo futuro y mejoras

En este algoritmo el nudo del for necesita ejecutarse para $1 \leq a \leq 2\sqrt{r}\log n$ para asegurar que el orden del tamaño del grupo G mencionado en el lema

3.3.7 es suficientemente grande ($> n^{2\sqrt{r}}$).

El límite superior para a podría ser mejorado si se pudiera demostrar que el conjunto de las $(x-a)$'s genera el grupo del tamaño requerido. Lo cual parece ser muy probable.

Se puede mejorar por mucho la complejidad a $\widetilde{O}(\log^3 n)$ si se pudiera demostrar la siguiente conjetura dada en [26] y verificarla para $r \leq 100$ y $n \leq 1010$ en [32]:

Conjetura 3.3.15 *Si r no divide a n y si*

$$(x-1)^n \equiv (x^n - 1) \mod (x^r - 1,\ n) \tag{3.4}$$

entonces cualquiera de las dos siguientes afirmaciones se cumplen, n es primo o $n^2 \equiv 1 \mod r$.

Si esta conjetura es verdadera, se puede modificar el algoritmo levemente a la primer búsqueda para un r que no divida a $n^2 - 1$. Tal r se puede encontrar en el rango $[2,\ 4\log n]$. Ésto es porque el producto de números primos menores que x es por lo menos e^x [22]. Posteriormente se puede probar si la congruencia (3.4) se cumple o no.

Verificando la congruencia (3.4) tomando el tiempo $\widetilde{O}(r(\log^2 n))$ y usando la Multiplicación Rápida de Fourier para la multiplicación nos daría una complejidad en el tiempo de $\widetilde{O}(\log^3 n)$.

3.3.7 Programación en Maple

El siguiente código corresponde a la programación en Maple del algoritmo AKS original:

```
esprimo:=proc(n) local log2n,pison,b,c,fc,r,mcd,cota,q::real,
pisoc,a,potencia1,potencia2,evaluadesigualdad,desigualdad,i,siesp,
noesp,t2;

  log2n:=log[2](n):
  pison:=floor(log2n):
```

```
 for b from 2 to pison do
    c:=n^(1/b):
    fc:=floor(c):
    if ((c-fc) = 0) then;
      RETURN(false)
    end if;
 end do:

 r:=3:

while (r<n) do;
 mcd:=gcd(n,r);
 if (mcd<>1) then;
   RETURN(false)
 end if;
 cota:=4*sqrt(r)*(log[2](n));
 for i from 2 to floor(sqrt(r)) do
    while (r = 0 mod i) do
         r:=r/i;
    end do;
 end do;
 if (i>=floor(sqrt(r))) then
   siesp:=1;
 else
   noesp:=0;
 end if;
 if (siesp = 1) then;
   t2:=r-1;
   for i from 2 to t2 do
      while (t2 = 0 mod i) do
           t2:=t2/i;
      end do;
   q:=i;
   end do;
   desigualdad:=q >= cota:
   evaluadesigualdad:=evalb(desigualdad):
     if (evaluadesigualdad=true) then;
       if ((n^((r-1)/q) mod r = 1 mod r)) then;
```

```
          break;
        end if;
      end if;
 end if;
 r:=r+2:
end do;

cota:=2*sqrt(r)*log[2](n):
pisoc:=floor(cota):

for a from 1 to pisoc do
   potencia1:=Powmod(x-a,n,x^r-1,x) mod n;
   potencia2:=Powmod(x^n-a,1,x^r-1,x) mod n;
   if (potencia1<>potencia2) then;
     RETURN(false)
   end if;
end do:
RETURN(true)
end proc;


final:=proc(t);
if (esprimo(t)=true) then
  print(‘ Es primo‘);
else
  print(‘ Es compuesto‘);
end if;
end proc;
```

Capítulo 4

Conclusión

Finalmente revisando el estado actual de AKS y algunas predicciones hechas. El problema de primalidad había sido tratado con teorías complicadas, tales como curvas elípticas, campos ciclotómicos, etc. Ha sido una gran sorpresa para muchos, que con métodos de álgebra básica se tenga una solución satisfactoria a este famoso problema. El algoritmo AKS, sin embargo, en su forma original [1] no es práctico, requiere un tiempo de ejecución $O(\log n^{10.5})$, mientras que el algoritmo más usado en la práctica (el de Miller-Rabin) tiene una complejidad en tiempo $O(\log n^3)$. Sin embargo, repetimos, como en casi todo nuevo invento hay una etapa de perfeccionamiento que casi siempre tiene buenos resultados. En el caso de AKS el primer ajuste lo hizo Lenstra, como ya se mencionó, y fue publicado por los mismos autores, mejorando el tiempo asintótico a $O(\log n^{6.5})$ [2]. Posteriormente se dan otros perfeccionamientos, algunos de ellos presentados por D. Berstein en [6], particularmente con el cálculo mejorado de $(x+a)^n$ en el anillo $(\mathbb{Z}_n[x]/(x-1)^r)$. También se puede usar el método de Berstein para detectar potencias perfectas, lo cual se describe en [6], en un tiempo de ejecución lineal. Otra propuesta más, fue hecha por P. Berrizbeitia y consiste en substituir la comprobación de la igualdad $(x+a)^n = (x^n+a) \mod (x^r-1)$ por la de $(mx+1)^n = (1+mx^n) \mod (x^{2^s}-a)$, para valores s, m adecuados. Esto reduce el número de evaluaciones polinomiales, lo cual, naturalmente es lo más deseado. La modificación da como resultado primos $n \equiv \pm 1 \mod 4$. Por otra parte, se han comparado los métodos AKS con el que utiliza curvas elípticas [15], y se ha propuesto combinar los dos métodos para hacer más eficiente la búsqueda [9]. Otras propuestas para mejorar el algoritmo se pueden consultar en [14, 17], así como análisis de su implementación en [7, 10]. Si se quiere ver la descripción de

AKS como una lectura general, se puede ver [8], y para una consulta más completa y detallada [11, 16]. Finalmente los autores del algoritmo AKS prevén que si se resuelven algunas pequeñas conjeturas, (si $n \neq 1 \mod r$ para algún $r > \log\log n$ y $(x-1)^n = (x^n-1) \mod (x^r-1,\ n)$, entonces n necesariamente es primo), AKS puede alcanzar un perfeccionamiento $O(\log n^3)$, lo que sería una muy buena noticia. Los tiempos prácticamente aceptables son $O(\log n^4)$, mas por el momento uno se conforma con un algoritmo de tiempo promedio $O(\log n^6)$.

Bibliografía

[1] Manindra Agrawal, Neeraj Kayal y Nitin Saxena, *Primes is in P*. Technical report, IIT Kanpur, 2002. Disponible en `http://www.cse.iitk.ac.in/news/primality.pdf`.

[2] Manindra Agrawal, Neeraj Kayal y Nitin Saxena, *Primes is in P*. revised version, IIT Kanpur, 2002. Disponible en`http://www.cse.iitk.ac.in/news/primality_v3.pdf`.

[3] T. Alexander y C. S. Tou, *AKS Algorithm*. Draft. University of Ottawa. Disponible en `http://padic.mathstat.uottawa.ca/~reports/AKS.pdf`.

[4] D. J. Bernstein, *Proving Primality After Agrawal-Kayal-Saxena*. University of Illinois at Chicago. Disponible en `http://.cr.yp.to/papers/aks.pdf`.

[5] D. J. Bernstein, *Detecting Perfect Powers In Essentially Linear Time*. Mathematics of Computation, University of Illinois at Chicago. Disponible en `http://.cr.yp.to/papers/powers.pdf`.

[6] D. J. Bernstein, H. W. Lenstra y J. Pila, *Detecting Perfect Powers by Factoring into Coprimes*. University of Illinois at Chicago. Disponible en `http://.cr.yp.to/lineartime/powers2-20050509.pdf`.

[7] V. Bhatt y G. K. Patra *Analysis on Implementation and its further improvements of AKS class algorithms*. Research Report CM 0307, Bangalore 560 037, India. Disponible en `http://www.cmmacs.ernet.in/cmmacs/Publications/resch_rep/rrcm0307.pdf`.

[8] F. Bornemann, *Primes is in P: A Breakthrough for Everyman*. Notices of the AMS, Mayo 2003, traducción de la versión alemana 2002. Disponible en `http://www.ams.org/notices/200305/fea-bornemann.pdf`.

[9] Q. Cheng, *Primality proving via one round in ECPP and one iteration in AKS*. The University of Oklahoma, Norman OK 73019. Disponible en `http://cr.yp.to/bib/2003/cheng.pdf`.

[10] R. Crandall y J.Papadopoulos *On the implementation of AKS-class primality test*. University of Maryland, Apple Computer, 18 marzo 2003. Disponible en `http://images.apple.com/acg/pdf/aks3.pdf`.

[11] A. Granville *It is easy to determine whether a given integer is prime*. Bulletin of the American Mathematical Society, Volumen 42, Número 1, páginas 3-38. Disponible en `http://www.mathematik.uni-wuerzburg.de/~mueller/Teach/granville.pdf`.

[12] M. J. Jacobson Jr. *An Exposition of the AKS Polynomial-Time Primality Test*. University of Calgary, 29 de Noviembre de 2002.

[13] A. Klappenecker *An Introduction to the AKS Primality Test*. Texas A&M University. Disponible en `http://faculty.cs.tamu.edu/klappi/629/aks.pdf`.

[14] M. Mačaj *Some remarks and questions about the AKS algorithm and related conjecture*. Comenius University, Slovakia. Disponible en `http://thales.doa.fmph.uniba.sk/macaj/aksremarks.pdf`.

[15] C. Rotella *An efficient implementation of the AKS polynomial time proving algorithm*. 29 de Abril de 2005. Disponible en `http://www-2.cs.cmu.edu/afs/cs.cmu.edu/user/mjs/ftp/thesis-05/rotella-thesis.pdf`.

[16] M. Smid *Primality testing in polynomial time*. Carleton University Ottawa, Ontario. Disponible en `http://www.scs.carleton.ca/~michiel/primes.ps.gz`.

[17] J. F. Voloch *Improvements to AKS*. University of Texas, Austin USA. Disponible en `http://www.ma.utexas.edu/users/voloch/Preprints/aks.pdf`.

[18] Gilbert Strang, cuarta edición *Álgebra lineal y sus aplicaciones*. Thomson, México.

[19] I. Vinográdov, *Fundamentos de la teoría de los números*. Editorial Mir. Moscu, 1987.

[20] M. Agrawal y S. Biswas, *Primality and identity testing via chinese remaindering*. In Proceedings of Annual IEEE Symposium on Foundations of Computer Science, páginas 202-209, 1999.

[21] L. M. Adleman y M. D. Huang, *Primality testing and two dimensional Abelian verieties over finite fields*. Lecture Notes in Mathematics, 1512, 1992.

[22] T. M. Apostol, *Introduction to Analytic Number Theory*. Springer-Verlag, 1997.

[23] L. M. Adleman, C. Pomerance, y R. S. Rumely, *On distinguishing prime numbers from composite numbers*. Ann. Math, 117:173-206, 1983.

[24] A. O. L. Atkin, *Lecture notes of a conference*. Manuscript, Agosto 1986.

[25] R. C. Baker y G. Harman, *The Brun-Titchmarsh Theorem on average*. Proceedings of a conference in Honor of Heini Halberstam, Volumen 1, páginas 39-103, 1996.

[26] Rajat Bhattacharjee y Prashant Pandey, *Primality testing*. Technical report, IIT Kanpur, 2001. Disponible en `http://www.cse.iitk.ac.in/research/btp2001/primality.html`.

[27] E. Fouvry, *Theoreme de Brun-Titchmarsh; application au theoreme de Fermat*. Invent. Math., 79:383-407, 1985.

[28] J. B. Fraleigh, *A First Course in Abstract Algebra*. Narosa, 1990.

[29] s. Goldwasser y J. Kilian, *Almost all primes can be quickly certified*. In Proceedings of Annual IEEE Symposium on Foundations of Computer Science, páginas 316-329, 1986.

[30] G. H. Hardy y J. E. Littlewood, *Some problems of Partitio Numerorum III: On the expression of a number as a sum of primes*. Acta Mathematica, 44:1-70, 1922.

[31] D. E Knuth, *The Art of Computer Programming*. Volumen 2, Seminumerical Algorithms, Addison Wesley, 1998.

[32] Neeraj Kayal y Nitin Saxena, *Towards a deterministic polynomial-time test*. Technical report, IIT Kanpur, 2002. Disponible en `http://www.cse.iitk.ac.in/research/btp2002/primality.html`.

[33] R. Lidl y H. Niederreiter, *Introduction of finite fields and their applications*. Cambridge University Press, 1986.

[34] G. L. Miller, *Riemann's hypothesis y tests for primality*. J. Comput. Sys. Sci., 13:300-317, 1976.

[35] M. O. Rabin, *Probabilistic algorithm for testing primality*. J. Number Theory, 12:128-138, 1980.

[36] R. Solovay y V. Strassen, *A fast Monte-Carlo test for primality*. SIAM Journal on Computing, 6:84-86, 1977.