UNIVERSIDADE ESTADUAL DO CEARÁ

CENTRO DE CIÊNCIAS E TECNOLOGIA

MESTRADO PROFISSIONAL EM MATEMÁTICA EM REDE NACIONAL

ED MASSEY MARTINS MENEZES FILHO

TESTES DE PRIMALIDADE

FORTALEZA – CEARÁ

2019

ED MASSEY MARTINS MENEZES FILHO

TESTES DE PRIMALIDADE

Dissertação apresentada ao Curso de Mestrado Profissional em Matemática em Rede Nacional do Centro de Ciências e Tecnologia da Universidade Estadual do Ceará, como requisito parcial à obtenção do título de mestre em Matemática. Área de Concentração: Teoria dos Números

Orientador: Prof. Dr. Tiago Caúla Ribeiro

FORTALEZA – CEARÁ

2019

Dados Internacionais de Catalogação na Publicação

Universidade Estadual do Ceará

Sistema de Bibliotecas

Menezes Filho, Ed Massey Martins.
Testes de primalidade [recurso eletrônico] / Ed Massey Martins Menezes Filho. - 2019.
1 CD-ROM: il.; 4 ¾ pol.

CD-ROM contendo o arquivo no formato PDF do trabalho acadêmico com 84 folhas, acondicionado em caixa de DVD Slim (19 x 14 cm x 7 mm).

Dissertação (mestrado profissional) - Universidade Estadual do Ceará, Centro de Ciências e Tecnologia, Mestrado Profissional em Matemática em Rede Nacional, Fortaleza, 2019.
Área de concentração: Matemática.
Orientação: Prof. Dr. Tiago Caúla Ribeiro.

1. Testes de primalidade. 2. Solovay e Strassen. 3. Miller-Rabin. 4. Crivo de Eratóstenes. 5. AKS. I. Título.

ED MASSEY MARTINS MENEZES FILHO

TESTES DE PRIMALIDADE

Dissertação apresentada ao Curso de Mestrado Profissional em Matemática em Rede Nacional do Centro de Ciências e Tecnologia da Universidade Estadual do Ceará, como requisito parcial à obtenção do título de mestre em Matemática. Área de Concentração: Teoria dos Números

Aprovada em: 23 de Agosto de 2019

BANCA EXAMINADORA

Prof. Dr. Tiago Caúla Ribeiro (Orientador)
Universidade Estadual do Ceará – UECE

Prof. Dr. João Montenegro de Miranda
Universidade Estadual do Ceará – UECE

Prof. Dr. Marcelo Ferreira de Melo
Universidade Federal do Ceará – UFC

Dedico esse trabalho aos apaixonados pela beleza da matemática pura.

AGRADECIMENTOS

À minha mãe Malena, pelo amor e fé inabaláveis.

Aos meus avós Marniele e Afonso, por acreditarem em meu potencial.

À minha esposa Emylie e a nosso filho Joaquim, pela alegria que me trazem todos os dias.

Aos meus irmãos Ana, Eduardo e João, que em todos os momentos estivemos juntos e sei que sempre estarão ao meu lado.

Ao meu orientador professor Tiago Caúla Ribeiro, por me conduzir e apresentar a tão belos temas para a pesquisa matemática.

Aos professores João Montenegro de Miranda e Marcelo Ferreira de Melo, por aceitarem o convite para participar da banca e pelas valiosas sugestões.

Aos professores da Pós-graduação em Matemática em Rede Nacional da UECE, em especial pela paciência e serenidade científica.

A todos os meus amigos do PROFMAT, pela amizade e equilíbrio gerado de nossas discussões.

"Nosso mundo ressoa com padrões. O crescente e minguante da lua. A mudança das estações. A estrutura celular microscópica de todos os seres vivos tem padrões. Talvez isso explique nossa fascinação por números primos que são os únicos sem padrão. Os números primos estão entre os fenômenos mais misteriosos da matemática."

(Manindra Agrawal)

RESUMO

Sabemos que primos são números naturais divisíveis apenas por 1 e ele mesmo. Mas quem foi que descobriu os números primos e com que intuito? Existem infinitos números primos? Como faço para testar se um número é primo? Onde estão presentes os números primos no nosso cotidiano? Os números primos são como os átomos que compõem todos os outros. Este trabalho envolve um dos assuntos mais belos da teoria dos números, os números primos. Começando com uma breve história dos números primos e sua utilização na criptografia; logo após apresentaremos algumas ferramentas de teoria dos números que serão usadas nos testes de primalidade. Apresentaremos alguns testes de primalidade e seus algoritmos computacionais, comparando qual é mais eficiente. O intuito desse trabalho é apresentar a importância dos números primos e com isso abrir uma reflexão de como devemos apresentá-los em sala de aula.

Palavras-chave: Testes de primalidade. Solovay e Strassen. Miller-Rabin. Crivo de Eratóstenes. AKS.

ABSTRACT

We know that cousins are natural numbers divisible only by 1 and itself. But who discovered the prime numbers and for what purpose? Are there infinite prime numbers? How do I test if a number is prime? Where are the prime numbers present in our daily lives? Prime numbers are like the atoms that make up all the others. This work involves one of the most beautiful subjects of number theory, the prime numbers. Beginning with a brief history of prime numbers and their use in cryptography; soon after we will present some number theory tools that will be used in primality tests. We will present some primality tests and their computational algorithms, comparing which is more efficient. The purpose of this paper is to present the importance of the prime numbers and with this to open a reflection of how we present them in the classroom.

Keywords: Primality tests. Solovay and Strassen. Miller-Rabin. Sieve of Eratosthenes. AKS.

SUMÁRIO

1 INTRODUÇÃO

Os números primos são caracterizados por serem divisíveis somente por 1 e por ele mesmo. Todo natural maior que 1 ou é primo ou composto pelo produto deles, por esse motivo os primos são chamados por muitos como os átomos dos números naturais. Muitos não compreendem a importância dos números primos, mas ao analisarmos a história da Matemática vemos que esses enigmáticos números são fundamentais para certos estudos e avanços tecnológicos.

1.1 HISTÓRIA

Tomando como base o estudado nas fontes (MERSENNE, 1996), (NOé, 2017) e (AKEL, 2016) para escrever o que se segue. Vamos aqui contar um pouco sobre a história dos números primos, da sua descoberta até sua utilização nos dias atuais.

O sistema decimal, formado pelos algarismos conforme utilizamos na atualidade, teve origem em trabalhos iniciados pelos hindus e pelos árabes. Mas antes de conhecer esse sistema de numeração, os matemáticos gregos da escola Pitagórica (500 a 300 a.C.) costumavam representar os números como pontos em forma geométrica. Os números eram distribuídos de maneira a formarem retângulos ou quadrados. Mas alguns números somente poderiam ser representados em uma linha, esses números eram chamados de primários, que traduzidos em latim recebiam o nome de primos. Os números que formavam retângulos e quadrados representavam os compostos, que de certa forma poderiam ser representados através de números primos. Essa utilização dos algarismos primos na representação dos números compostos é importante nas questões envolvendo simplificação de radicais e extração da raiz quadrada de um número. Além dessa utilização, os pitagóricos acreditavam que os números primos tinham poderes místicos.

Por volta de 300 a.C., o livro *Os Elementos* de Euclides apresentou vários resultados importantes sobre números primos. Euclides provou em um dos seus livros que existem infinitos números primos. Em outra parte do livro foi demonstrado o *Teorema Fundamental da Aritmética*, onde fala que qualquer número natural pode ser representado de forma essencialmente única como produto de primos.

Com o passar dos anos, e sabendo que existem infinitos primos, foi-se questionado como fazer para descobrir se um número grande é ou não primo. Então por volta de 200 a.C. tivemos o primeiro teste de primalidade registrado chamado de *Crivo de Eratóstenes* que será explicado mais à frente em detalhes.

Após os gregos, houve um longo tempo sem avanços na história dos números primos, esse período é denominado por muitos como *Idade das Trevas*. Os desenvolvimentos mais importantes só vieram ocorrer novamente no século XVII. Em 1640, Pierre de Fermat apresentou o *Pequeno Teorema de Fermat* que será usado como base para vários testes de primalidade. O monge Marin Mersenne, um contemporâneo de Fermat, avançou no estudo dos primos da forma $2^p - 1$ para p primo, chamados de *Primos de Mersenne*.

Euler, um gênio que dedicou sua vida à Matemática, mostrou que a série infinita $\frac{1}{2} + \frac{1}{3} + \frac{1}{5}...$ da soma dos inversos dos números primos é divergente. No início do século XIX, Legendre e Gauss conjecturaram de forma independente de que quando n tende ao infinito, o número de primos menores ou iguais a n é assintótica a função $\frac{n}{\ln n}$.

Para descobrir se um número é primo usamos testes de primalidade que se dividem em probabilísticos e determinísticos. Vamos nesse trabalho explicar em detalhes os testes de primalidade probabilísticos *Solovay e Strassen* e *Miller-Rabin* e os testes determinísticos *Crivo de Eratóstenes* e *AKS*.

1.2 PROJETO GIMPS

Os últimos números primos descobertos foram encontrados em um projeto voluntário chamando GIMPS (Great Internet Mersenne Prime Search). Inclusive você pode baixar o aplicativo deles e ajudá-los no processamento dos novos números. No dia 21 de dezembro de 2018 foi descoberto o maior número primo conhecido, $2^{82589933} - 1$, possuindo 24862048 dígitos. Além da busca por números primos o projeto é uma boa fonte de conhecimento e fatos curiosos sobre números primos.

1.3 CRIPTOGRAFIA

A criptografia é a técnica de transformar mensagens em um padrão não compreensível para que apenas o destinatário, através de um conhecimento privilegiado, possa retorná-las ao seu estado original, assim, protegendo-as de agentes não autorizados. Este conhecimento privilegiado é chamado de chave. É uma forma de escrita secreta.

Por muitos anos, reis, rainhas e generais confiavam o comando e governança de seus países a eficientes métodos de comunicação. Ao mesmo tempo, todos tinham consciência das consequências de suas mensagens serem lidas por pessoas não autorizadas, revelando segredos valiosos a nações rivais. Foi esse risco de interceptações inimigas que motivou o desenvolvimento

de códigos e cifras. A história da criptografia é a história das batalhas entre criadores de código e quebradores de código.

A construção de códigos seguros de criptografia é uma tarefa muito difícil, pois antes o remetente e destinatário compartilhavam a mesma chave de decodificação e os códigos eram em parte apenas embaralhamentos de caracteres. Podemos exemplificar com um código que troca as consoantes pela consoante duas casas a frente no alfabeto, sendo assim a palavra "matemática" seria escrita como "paxepáxifa", mas é facilmente decodificada. Um exemplo muito famoso é o caso do matemático Alan Turing que na Segunda Guerra Mundial conseguiu interceptar e decodificar as conversas das frotas navais da Alemanha.

O tempo passou e novos métodos de criptografia foram surgindo e hoje o mais utilizado é o método RSA. Os computadores têm dificuldade de fatoração e facilidade de multiplicação de números grandes e é nesse fato que o RSA se ancora, então ele efetua o produto de dois primos grandes, cria uma chave pública e uma chave privada. Por exemplo, eu chego e divulgo minha chave pública e qualquer pessoa poderá enviar-me uma mensagem codificada, mas só eu através da chave privada terei acesso ao conteúdo da mensagem. Esse método será explicado em detalhes no capítulo 3.

A segurança dos métodos utilizados atualmente para criptografia baseia-se na dificuldade computacional da fatoração de números grandes, proporcionando um bom nível de segurança. Com o crescimento do poder computacional poderá haver falhas de segurança. Sabendo desse fato, diversos núcleos de pesquisa vêm buscando métodos alternativos para substituir as técnicas existentes atualmente. A técnica que se mostra mais promissora é a *criptografia quântica*.

A *criptografia quântica* se baseia no princípio físico da mecânica quântica da dualidade da matéria para garantir a comunicação segura. Com ela, remetente e destinatário podem criar e compartilhar uma chave secreta para criptografar e descriptografar suas mensagens. Se algum intruso interceptar a mensagem e tentar descriptografá-la utilizando uma chave diferente da correta, a mensagem sofrerá alteração; quando o destinatário receber a mensagem, perceberá a tentativa de interceptação. O interceptador além de não conseguir ler a mensagem deixará ciente os envolvidos de que aquele efetuou uma tentativa. Bem mais seguro, mas isso é um assunto para um futuro não tão distante.

1.4 MOTIVAÇÃO

O que me motivou a escolher esse tema foi o capítulo 7 do livro (ROUSSEAU; SAINT-AUBIN, 2015) que fala sobre criptografia de chave pública, sua larga aplicabilidade e o fato de não ser dado devida importância em sala de aula. Escutamos muitos alunos perguntando: "onde vou usar isso na minha vida?", "pra que serve isso mesmo?". Quanto mais mostrarmos aplicabilidades e interligação com outras competências do aprendizado, mais fácil será ensinar. Mostrar que a Matemática está em todo canto e que o aluno precisa aprender para vida e não só para passar na prova, é o que realmente importa e o dever de todo professor.

1.5 OBJETIVOS

O objetivo desse trabalho é mostrar o embasamento teórico de alguns testes de primalidade, programá-los em linguagem $C++$ e testar sua eficiência.

2 FUNDAMENTAÇÃO TEÓRICA

Tendo como embasamento teórico os livros (SANTOS, 1998), (MARTINEZ, 2015), (GALLIER; QUAINTANCE, 2019), (HEFEZ, 2016), (NASCIMENTO; FEITOSA, 2009), (DIETZFELBINGER, 2004) e (MAIER, 2005). Apresento nesse capítulo algumas ferramentas matemáticas que serão usadas nas provas dos teoremas subsequentes.

O máximo divisor comum de a e b é denotado por (a,b). Quando $(a,b) = 1$ temos que não existe divisor diferente de 1 em comum entre a e b, então dizemos que a e b são primos entre si.

Denotamos de $a!$ o produto $1.2.3...a$, chamamos de a fatorial. Convencionamos que $0! = 1$.

Definição 2.0.1 (Número Binômio) *Sendo a e n números naturais temos*

$$\binom{n}{a} = \frac{n!}{a!(n-a)!}$$

Quando $a > n$ convencionamos $\binom{n}{a} = 0$.

2.1 ARITMÉTICA MODULAR

Sejam $n, a \in \mathbb{Z}$. Dizemos que n divide a, que equivale a dizer que n é um divisor de a ou que a é um múltiplo de n e denotado por $n|a$, se e somente se existe um inteiro m tal que $a = nm$. Quando n não divide a representamos por $n \nmid a$.

Sendo $a,b,n \in \mathbb{Z}$. Falamos que a é congruente a b módulo n, quando $n \mid a - b$. Denotada por:

$$a \equiv b \mod n$$

Equivale a dizer que a e b deixam o mesmo resto quando divididos por n, pois se $a = qn + r$, com r sendo o resto da divisão e q o quociente. Como $n \mid a - b \Leftrightarrow a - b = mn$, para um $m \in \mathbb{Z}$. Temos:

$$mn = (qn + r) - b \Leftrightarrow b = qn + r - mn = (q - m)n + r$$

Como $n|(q-m)n$ e $0 \leq r \leq n-1$ então podemos concluir que b deixa resto r quando dividido por n.

2.1.1 Propriedades Modulares

Para quaisquer $a,b,c,d,n \in \mathbb{Z}$ são validas as propriedades que se seguem:

1. *Reflexividade*: $a \equiv a \mod n$.
2. *Simetria*: $a \equiv b \mod n$, então $b \equiv a \mod n$.
3. *Transitividade*: $a \equiv b \mod n$ e $b \equiv c \mod n$, então $a \equiv c \mod n$.
4. *Adição e subtração compatíveis*: Pode somar e subtrair os membros:

$$\begin{cases} a \equiv b \mod n \\ c \equiv d \mod n \end{cases} \Longrightarrow \begin{cases} a+c \equiv b+d \mod n \\ a-c \equiv b-d \mod n \end{cases}$$

   consequentemente temos $ka \equiv kb \mod n$, $\forall\, k \in \mathbb{N}$.

5. *Produto compatível*: Pode multiplicar os membros:

$$\begin{cases} a \equiv b \mod n \\ c \equiv d \mod n \end{cases} \Longrightarrow ac \equiv bd \mod n$$

   consequentemente temos $a^k \equiv b^k \mod n$, $\forall\, k \in \mathbb{N}$.

6. *Simplificação*: Se $mdc(c,n) = 1$, então

$$ac \equiv bc \mod n \Leftrightarrow a \equiv b \mod n$$

Lema 2.1.1 *Sendo $n \in \mathbb{Z}^+$, então n é primo se, e semente se*

$$\binom{n}{k} \equiv 0 \mod n, \text{ para todo } 0 < k < n.$$

Demonstração 2.1.1 *Se n é primo, então $n \nmid k!$ nem $n \nmid (n-k)!$,para $0 < k < n$, mas $n \mid n!$. Portanto o fator n que aparece no numerador do quociente não é cancelado pelo denominador e com isso podemos afirmar que*

$$n \mid \binom{n}{k}.$$

Se n é composto, seja p um divisor primo de n. Digamos que p^α é a maior potência de p que divide n, ou seja, $n = p^\alpha.b$ e $p \nmid b$. Temos então que

$$\binom{n}{p} = \frac{n.(n-1)!}{p!(n-p)!} = p^{\alpha-1}.b.\frac{(n-1)!}{(p-1)!(n-p)!}.$$

Por tanto para que $n \mid \binom{n}{p}$, precisamos que $p \mid \frac{(n-1)!}{(p-1)!(n-p)!}$.

Mas

$$\frac{(n-1)!}{(p-1)!(n-p)!} = \frac{(n-1)(n-2)...(n-(p-1))(n-p)!}{(p-1)!(n-p)!} = \frac{(n-1)(n-2)...(n-(p-1))}{(p-1)!}.$$

Como

$$p \mid n \Rightarrow p \nmid (n-1)(n-2)...(n-(p-1)) \Rightarrow p \nmid \frac{(n-1)!}{(p-1)!(n-p)!} \Rightarrow n \nmid \binom{n}{p}.$$

Sendo assim se n composto teremos números binomiais que não serão divisíveis por n, concluindo a prova.

□

2.2 O PEQUENO TEOREMA DE FERMAT E A FUNÇÃO DE EULER

Teorema 2.2.1 (Pequeno teorema de Fermat) *Se p é primo e $a \in \mathbb{Z}$, então p divide $a^p - a$. Observação*

$$p \mid a^p - a \Leftrightarrow a^p \equiv a \mod p$$

Demonstração 2.2.1 *Vamos primeiro provar por indução para os casos onde $a \geq 0$. Para $a = 0$ e $a = 1$ é imediato. Agora supondo válido para um certo a, onde $a^p \equiv a \mod p$. Efetuando o desenvolvimento binomial*

$$(a+1)^p = a^p + \binom{p}{1}a^{p-1} + \binom{p}{2}a^{p-2} + ... + \binom{p}{p-1}a + 1$$

Para $k \in \{1,2,...,p-1\}$ temos pelo lema 2.1.1 que $p \mid \binom{p}{k}$, então

$$(a+1)^p \equiv a^p + 1 \mod p$$

mas pela hipótese de indução temos que $a^p \equiv a \mod p$, então

$$(a+1)^p \equiv a^p + 1 \equiv a + 1 \mod p$$

Agora só nos resta provar para $a < 0$. Tomemos $a = -b$, então

$$a^p = (-1)^p b^p \equiv (-1)^p b = \begin{cases} -b = a \mod p \text{ se } p \neq 2 \\ b = -a \equiv a \mod 2 \end{cases}$$

Está provado que para todo a inteiro $a^p \equiv a \mod p$.

□

Observação, quando $(a,p) = 1$ temos que $p \nmid a$, então como

$$p \mid a^p - a = a(a^{p-1} - 1) \Rightarrow p \mid a^{p-1} - 1 \Leftrightarrow a^{p-1} \equiv 1 \mod p$$

Definição 2.2.1 (Função Totiente de Euler) *$\varphi(n)$ representa a quantidade de números i do conjunto $\{1,2,...,n-1\}$ que são relativamente primos com n, ou seja, $(n,i)=1$.*

Por convenção temos $\varphi(1)=1$. Por definição é fácil verificar que se p é primo então $\varphi(p)=p-1$.

Proposição 2.2.2 *Tomando p,q números primos naturais temos que*

$$\varphi(p.q)=(p-1).(q-1)$$

Demonstração 2.2.2 *Precisamos verificar quantos inteiros do conjunto$\{1,2,...,pq-1\}$ que são relativamente primo com $p.q$. Os únicos inteiros que não são relativamente primos com $p.q$ são os múltiplos de p que são $P=\{p,2p,...(q-1).p\}$ e os de q que são $Q=\{q,2q,...,(p-1).q\}$. Note que $P\cap Q=0$. Então:*

$$\varphi(p.q)=(p.q-1)-(p-1)-(q-1)=p.q-p-q+1=(p-1).(q-1)$$

□

O teorema a seguir é uma generalização do Pequeno teorema de Fermat.

Teorema 2.2.3 (Euler) *Sejam m e n dois inteiros, com $n>0$ e $(n,m)=1$. Então:*

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

O único caso relevante para esse trabalho será quando $n=p.q$ com p e q inteiros e primos distintos. Então não será necessário provar o teorema de Euler por completo apenas verificar que como $\varphi(n)=(p-1).(q-1)$, utilizando 2.2.1, temos

$$\begin{cases} m^{p-1}\equiv 1 \mod p \\ m^{q-1}\equiv 1 \mod q \end{cases} \Rightarrow \begin{cases} (m^{p-1})^{q-1}=m^{\varphi(n)}\equiv 1 \mod p \\ (m^{q-1})^{p-1}=m^{\varphi(n)}\equiv 1 \mod q \end{cases} \Rightarrow \begin{cases} p|m^{\varphi(n)}-1 \\ q|m^{\varphi(n)}-1 \end{cases}$$

Como $(p,q)=1$ temos que $p.q|m^{\varphi(n)}-1$, ou seja, $n|m^{\varphi(n)}-1$ e por tanto

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

2.3 MÁXIMO DIVISOR COMUM

Denotemos por $e\mathbb{Z}=\{ed|\forall d\in\mathbb{Z}\}$ e $a\mathbb{Z}+b\mathbb{Z}=\{ax+by|\forall x,y\in\mathbb{Z}\}$.

Teorema 2.3.1 *Sejam $a, b \in \mathbb{Z}$, não ambos nulos. Se d é o menor elemento natural do conjunto $a\mathbb{Z} + b\mathbb{Z}$, então*

1. $d = (a,b)$;
2. $a\mathbb{Z} + b\mathbb{Z} = d\mathbb{Z}$.

Demonstração 2.3.1 *Temos que*

$$\begin{cases} (a,b) \mid ax \\ (a,b) \mid by \end{cases} \Rightarrow (a,b) \mid ax + by \Rightarrow (a,b) \mid d$$

Vamos agora provar que d divide todos os elementos de $a\mathbb{Z} + b\mathbb{Z}$. Supondo por contradição que para um $w \in a\mathbb{Z} + b\mathbb{Z}$ temos $d \nmid w$, logo pela divisão euclidiana

$$w = dq + r, \text{ com } 0 < r < d.$$

Sendo $d = ax_0 + by_0$ e $w = ax + by$, com $x_0, y_0, x, y \in \mathbb{Z}$, temos que

$$r = w - dq = ax + by - (ax_0 + by_0)q = a(x - qx_0) + b(y - qy_0) \in a\mathbb{Z} + b\mathbb{Z} \cap \mathbb{N}$$

Absurdo, logo $r = 0$ e $d \mid w$, $\forall w \in a\mathbb{Z} + b\mathbb{Z}$. Então podemos afirmar que $d \mid a$ e $d \mid b$, sendo assim $d \mid (a,b)$. Como $d \mid (a,b)$ e $(a,b) \mid d$, então $d = (a,b)$.

Agora vamos provar que $a\mathbb{Z} + b\mathbb{Z} = d\mathbb{Z}$. Como todos os elementos de $a\mathbb{Z} + b\mathbb{Z}$ são divisíveis por d, então $a\mathbb{Z} + b\mathbb{Z} \subset d\mathbb{Z}$. Por outro lado, para todo $ed \in d\mathbb{Z}$, temos que

$$ed = e(ax_0 + by_0) = a(ex_0) + b(ey_0) \in a\mathbb{Z} + b\mathbb{Z}$$

portanto $d\mathbb{Z} \subset a\mathbb{Z} + b\mathbb{Z}$. Concluímos que $a\mathbb{Z} + b\mathbb{Z} = d\mathbb{Z}$.

□

Proposição 2.3.2 *Dois números inteiros a e b são primos entre si se, e somente se, existem números inteiros x e y tais que $ax + by = 1$.*

Demonstração 2.3.2 *Supondo que $(a,b) = 1$, temos pelo teorema 2.3.1 que existem números inteiros x e y tais que $ax + by = (a,b) = 1$.*

Reciprocamente, supondo que existem números inteiros x e y tais que $ax + by = 1$, então

$$(a,b) \mid ax + by \Rightarrow (a,b) \mid 1 \Rightarrow (a,b) = 1$$

□

Proposição 2.3.3 *Sejam $a,n \in \mathbb{Z}$ com $n > 1$. A congruência $ax \equiv 1 \mod n$ possui solução se, e somente se, $(a,n) = 1$. Além disso se $x_0, x \in \mathbb{Z}$ são soluções, então $x \equiv x_0 \mod n$.*

Demonstração 2.3.3 *Se x_0 é solução, então $n \mid ax_0 - 1$, ou seja, existe um $y_0 \in \mathbb{Z}$ tal que $ny_0 = ax_0 - 1$, então $1 = ax_0 - ny_0$. Pela proposição 2.3.2 isso ocorre se, e somente se, $(a,b) = 1$.*

Se x_0 e x são soluções da congruência, então $ax \equiv ax_0 \mod n$. Pela propriedade modular de simplificação da seção 2.1.1, temos que $x \equiv x_0 \mod n$.

□

Teorema 2.3.4 (Wilson) *Para qualquer primo p temos que $(p-1)! \equiv -1 \mod p$.*

Demonstração 2.3.4 *O teorema é válido para $p = 2$ e $p = 3$. Supondo $p > 3$ primo. Para todo $i \in \{1,...,p-1\}$, pela proposição 2.3.3, a congruência $ix \equiv 1 \mod p$ possui uma única solução módulo p, ou seja, dado $i \in \{1,...,p-1\}$ existe um único $j \in \{1,...,p-1\}$ tal que $ij \equiv 1 \mod p$. Por outro lado, se $i \in \{1,...,p-1\}$ é tal que $i^2 \equiv 1 \mod p$, então $p \mid i^2 - 1$, o que equivale a $p \mid i-1$ ou $p \mid i+1$, o que só ocorre se $i = 1$ ou $i = p-1$. Logo,*

$$2...(p-2) \equiv 1 \mod p$$

e portanto,

$$1.2...(p-2)(p-1) \equiv (p-1)! \equiv -1 \mod p.$$

□

Observemos que as propriedades modulares de reflexividade, simetria e transitividade da seção 2.1.1 nos dizem que congruência módulo n é uma relação de equivalência em $\mathbb{Z}$.

Definição 2.3.1 (Classe de Equivalência) *Dado n um inteiro positivo, para cada $a \in \mathbb{Z}$, denotamos a classe de equivalência de a módulo n por $\overline{a} = \{r \in \mathbb{Z} | r \equiv a \mod n\}$.*

Definição 2.3.2 ($\mathbb{Z}_n$) *É o conjunto quociente de $\mathbb{Z}$ pela relação de congruência módulo n. Assim, $\mathbb{Z}_n = \{\overline{a} | a \in \mathbb{Z}\}$.*

Definição 2.3.3 (Sistema Completo de Restos) *Dizemos que um conjunto $\{a_1, a_2, a_3, ..., a_n\}$ de n elementos inteiros forma um sistema completo de restos módulo n, indicado por SCR, se os a_i representam todas as classes de equivalência módulo n. Por definição então $\mathbb{Z}_n$ é um SCR. Podemos afirmar que $a_1, a_2, ..., a_n$ formam um SCR módulo n se, e somente se, $a_i \equiv a_j \mod n$ implicando em $i = j$. Exemplificando, temos que $0, 1, 2, ..., n-1$ é um SCR módulo n.*

Definição 2.3.4 (Sistema Reduzido de Resíduos) *É chamado um sistema reduzido de resíduos, SRR, módulo n qualquer conjunto com* $\varphi(n)$ *inteiros primos com n e que são incongruentes entre si módulo n. Exemplificando, temos que* $1,3,7,9$ *é um SRR módulo* 10.

2.4 RESÍDUOS QUADRÁTICOS

Definição 2.4.1 *Seja p um primo impar e* $b \in \mathbb{Z}$ *um inteiro qualquer.*

$$x^2 \equiv b \mod p$$

Se a equação acima admite solução (se $\overline{b}$ *é um quadrado perfeito em* $\mathbb{Z}_p$*) então dizemos que b é um resíduo ou resto quadrático módulo p.*

Proposição 2.4.1 *Existe exatamente* $\frac{p+1}{2}$ *resíduos quadráticos módulo p, que equivale a:*

$$0^2, 1^2, 2^2, ..., \left(\frac{p-1}{2}\right)^2 \mod p$$

Demonstração 2.4.1 *Todo inteiro é congruente a* $\pm c$ *módulo p, com* $0 \leq c \leq \frac{p-1}{2}$*. Módulo p estes números são todos distintos, de fato, temos que:*

$$\begin{aligned} c^2 \equiv d^2 \mod p &\implies p|(c-d)(c+d) \\ p|c-d \text{ ou } p|c+d &\iff c \equiv \pm d \mod p \end{aligned}$$

Como $0 \leq c,d \leq \frac{p-1}{2} \implies 0 \leq c+d \leq p-1$*, então* $c-d=0$ *ou* $c+d=0$*, sendo assim podemos afirmar que a única possibilidade é* $c=d$.

□

Definição 2.4.2 (Símbolo de Legendre) *Para simplificar notação definimos o Símbolo de Legendre. Seja p um primo ímpar e a um inteiro qualquer, então:*

$$\left(\frac{a}{p}\right) = \begin{cases} 0 & \text{se } p|a \\ 1 & \text{se } a \text{ é resíduo quadrático de } p \\ -l & \text{se } a \text{ não é resíduo quadrático de } p \end{cases}$$

O Símbolo de Jacobi representa uma generalização do Símbolo de Legendre e segue sua definição.

Definição 2.4.3 (Símbolo de Jacobi) *Para um inteiro positivo a relativamente primo com o inteiro ímpar $n = p_1^{\alpha_1} p_2^{\alpha_2} ... p_k^{\alpha_k}$ o Símbolo de Jacobi, denotado por $(\frac{a}{n})$, é dado por:*

$$\left(\frac{a}{n}\right) = \left(\frac{a}{p_1^{\alpha_1} p_2^{\alpha_2} ... p_k^{\alpha_k}}\right) = \left(\frac{a}{p_1}\right)^{\alpha_1} \left(\frac{a}{p_2}\right)^{\alpha_2} ... \left(\frac{a}{p_k}\right)^{\alpha_k}$$

No lado direito da igualdade temos os Símbolo de Legendre.

Proposição 2.4.2 (Critério de Euler) *Se p um primo ímpar e a um inteiro qualquer, temos:*

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

Demonstração 2.4.2 *Para $p|a$ o resultado é imediato, então supondo $p \nmid a$. Pelo Pequeno Teorema de Fermat 2.2.1 temos que $a^{p-1} \equiv 1 \mod p$, então:*

Sendo a um resíduo quadrático de p então $\left(\frac{a}{p}\right) = 1$, que equivale a dizer que existe um x inteiro tal que $a \equiv x^2 \mod p$.

$$a \equiv x^2 \mod p \Longleftrightarrow a^{\frac{p-1}{2}} \equiv (x^2)^{\frac{p-1}{2}} \mod p \Longleftrightarrow a^{\frac{p-1}{2}} \equiv x^{p-1} \equiv 1 \mod p$$

Finalmente, só nos resta analisar quando $\left(\frac{a}{p}\right) = -1$. Como sabemos que $\forall i \in \{1,2,...,p-1\}, \exists! j \in \{1,2,...,p-1\}$ tal que $ij \equiv a \mod p$ e podemos afirmar que $i \neq j$, pois caso contrário a seria um resíduo quadrático. Agora dividindo o conjunto $\{1,2,...,p-1\}$ em pares cujo o produto é congruentes a a módulo p teremos $\frac{p-1}{2}$ pares, agora efetuando o produto de todos os pares temos

$$1.2...(p-1) = (p-1)! \equiv a^{\frac{p-1}{2}} \mod p$$

Mas pelo Teorema de Wilson 2.3.4 temos que $(p-1)! \equiv -1 \mod p$, portanto $\left(\frac{a}{p}\right) \equiv a^{\frac{p-1}{2}} \equiv (p-1)! \equiv -1 \mod p$.

□

Proposição 2.4.3 (Propriedades do Símbolo de Legendre e Jacobi) *Sejam n, m inteiros ímpares maiores que 2 e a, b inteiros.*

1. $a \equiv b \mod n \Rightarrow \left(\frac{a}{n}\right) = \left(\frac{b}{n}\right)$
2. $\left(\frac{ab}{n}\right) = \left(\frac{a}{n}\right)\left(\frac{b}{n}\right)$
3. $\left(\frac{a}{nm}\right) = \left(\frac{a}{n}\right)\left(\frac{a}{m}\right)$

Demonstração 2.4.3 .

1. *Se n é primo, então o resultado se segue da definição. Se n é composto com a decomposição em primos* $n = p_1^{\alpha_1} p_2^{\alpha_2} ... p_k^{\alpha_k}$ *temos que* $\forall\, i \in \{1,2,...,k\}$

$$a \equiv b \mod n \Rightarrow a \equiv b \mod p_i \Rightarrow \left(\frac{a}{p_i}\right) = \left(\frac{b}{p_i}\right)$$

$$\left(\frac{a}{p_i}\right)^{\alpha_i} = \left(\frac{b}{p_i}\right)^{\alpha_i} \Rightarrow \left(\frac{a}{p_1}\right)^{\alpha_1} \left(\frac{a}{p_2}\right)^{\alpha_2} ... \left(\frac{a}{p_k}\right)^{\alpha_k} = \left(\frac{b}{p_1}\right)^{\alpha_1} \left(\frac{b}{p_2}\right)^{\alpha_2} ... \left(\frac{b}{p_k}\right)^{\alpha_k}$$

2. *Pelo Critério de Euler 2.4.2, se* $n \mid ab$ *o resultado é imediato. Agora estudando os casos onde* $n \nmid ab$*: para n primo temos*

$$\left(\frac{ab}{n}\right) \equiv (ab)^{\frac{n-1}{2}} \equiv a^{\frac{n-1}{2}} b^{\frac{n-1}{2}} \equiv \left(\frac{a}{n}\right)\left(\frac{b}{n}\right) \mod n$$

Como ambos os lados da congruência são iguais a ± 1*, então*

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

Para n composto e com decomposição em primos $n = p_1^{\alpha_1} p_2^{\alpha_2} ... p_k^{\alpha_k}$ *temos que* $\forall\, i \in \{1,2,...,k\}$

$$\left(\frac{ab}{p_i}\right) = \left(\frac{a}{p_i}\right)\left(\frac{b}{p_i}\right) \Rightarrow \left(\frac{ab}{p_i}\right)^{\alpha_i} = \left(\frac{a}{p_i}\right)^{\alpha_i} \left(\frac{b}{p_i}\right)^{\alpha_i}$$

$$\left(\frac{ab}{p_1}\right)^{\alpha_1} \left(\frac{ab}{p_2}\right)^{\alpha_2} ... \left(\frac{ab}{p_k}\right)^{\alpha_k} = \left(\frac{a}{p_1}\right)^{\alpha_1} \left(\frac{b}{p_1}\right)^{\alpha_1} \left(\frac{a}{p_2}\right)^{\alpha_2} \left(\frac{b}{p_2}\right)^{\alpha_2} ... \left(\frac{a}{p_k}\right)^{\alpha_k} \left(\frac{b}{p_k}\right)^{\alpha_k}$$

3. *É imediato pela definição do Símbolo de Jacobi 2.4.3.*

□

2.4.1 Lei da Reciprocidade Quadrática

1. Sejam n e m números inteiros ímpares e primos entre si. Então podemos afirmar que

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

2. Seja n um inteiro ímpar. Então podemos afirmar que:

$$\left(\frac{2}{n}\right) = (-1)^{\frac{n^2-1}{8}} = \begin{cases} 1 & \text{se } n \equiv \pm 1 \mod 8 \\ -1 & \text{se } n \equiv \pm 3 \mod 8 \end{cases}$$

Para demonstrar a lei de reciprocidade quadrática devemos aprender primeiro o Lema de Gauβ.

Lema 2.4.1 (Gauβ) *Seja p um primo ímpar e a um inteiro tal que $(a,p)=1$. Seja $\lambda_p(a)$ o número de elementos do conjunto $\{a,2a,3a,...,\frac{p-1}{2}a\}$ tais que seu resto módulo p é maior do que $\frac{p-1}{2}$. Então $\left(\frac{a}{p}\right)=(-1)^{\lambda_p(a)}$.*

Demonstração 2.4.4 (Gauβ) *A ideia é imitar a prova do pequeno teorema de Fermat 2.2.1. Como o conjunto $\{\pm1,\pm2,...,\pm\frac{p-1}{2}\}$ é um SRR 2.3.4 de invertíveis módulo p, para cada $j=1,2,...,\frac{p-1}{2}$ podemos escrever $aj\equiv\varepsilon_j m_j \mod p$ com $\varepsilon_j\in\{-1,1\}$ e $m_j\in\{1,2,...,\frac{p-1}{2}\}$.*

Temos que se $i\neq j$ então $m_i\neq m_j$ donde $\{m_1,m_2,...,m_{\frac{p-1}{2}}\}=\{1,2,...,\frac{p-1}{2}\}$. De fato, se $m_i=m_j$ temos $ai\equiv aj \mod p$ ou $ai\equiv -aj \mod p$; como a é invertível módulo p e $0<i,j\leq\frac{p-1}{2}$, temos que a primeira possibilidade implica $i=j$ e a segunda é impossível. Assim, multiplicando as congruências $aj\equiv\varepsilon_j m_j \mod p$, obtemos

$$\begin{aligned}
&(1a)(2a)...(\tfrac{p-1}{2}a)\equiv\varepsilon_1\varepsilon_2...\varepsilon_{\frac{p-1}{2}}m_1m_2...m_{\frac{p-1}{2}} \mod p\\
\iff\quad & a^{\frac{p-1}{2}}\left(\tfrac{p-1}{2}\right)!\equiv\varepsilon_1\varepsilon_2...\varepsilon_{\frac{p-1}{2}}\left(\tfrac{p-1}{2}\right)! \mod p\\
\iff\quad & \left(\frac{a}{p}\right)\equiv\varepsilon_1\varepsilon_2...\varepsilon_{\frac{p-1}{2}} \mod p
\end{aligned}$$

Donde $\left(\frac{a}{p}\right)=\varepsilon_1\varepsilon_2...\varepsilon_{\frac{p-1}{2}}$, pois ambos os lados $\in\{-1,1\}$.

Como por definição $\varepsilon_j=-1$ equivale a dizer que o resto de aj por p é maior do que $\frac{p-1}{2}$, podemos concluir $\left(\frac{a}{p}\right)=(-1)^{\lambda_p(a)}$.

□

Demonstração 2.4.5 (Lei da Reciprocidade Quadrática) *Vamos primeiro provar o item 2 onde fala que $\left(\frac{2}{n}\right)=(-1)^{\frac{n^2-1}{8}}$ para n primo usando o lema de Gauβ 2.4.1. Se n for da forma $4k+1$ para um $k\in\mathbb{Z}$ temos que $\frac{n-1}{2}=2k$. Então no conjunto $\{2,4,...,\frac{n-1}{2},...,n-1\}$ temos k elementos maiores do que $\frac{n-1}{2}$, sendo assim*

$$\left(\frac{2}{n}\right)=(-1)^k=((-1)^k)^{2k+1}=(-1)^{\frac{n^2-1}{8}}=\begin{cases}1 & \text{, se } n\equiv 1 \mod 8\\ -1 & \text{, se } n\equiv 5 \mod 8\end{cases}$$

Se n for da forma $4k+3$ temos que $\frac{n-1}{2}=2k+1$. Então no conjunto $\{2,4,...,\frac{n-3}{2},\frac{n+1}{2},...,n-1\}$ temos $k+1$ elementos maiores do que $\frac{n-1}{2}$, sendo assim

$$\left(\frac{2}{n}\right)=(-1)^{k+1}=((-1)^{k+1})^{2k+1}=(-1)^{\frac{n^2-1}{8}}=\begin{cases}1 & \text{, se } n\equiv 7 \mod 8\\ -1 & \text{, se } n\equiv 3 \mod 8\end{cases}$$

Já sabemos que a igualdade é válida para n primo, nos resta provar para n composto. Então, por indução, vamos supor que é válida para k, w inteiros ímpares, então tomando $n = kw$ temos

$$\frac{n^2-1}{8} = \frac{(k^2-1)(w^2-1)+k^2+w^2-2}{8} = \frac{(k-1)(k+1)(w-1)(w+1)}{8} + \frac{k^2-1}{8} + \frac{w^2-1}{8},$$

fácil ver que $\frac{(k-1)(k+1)(w-1)(w+1)}{8} \equiv 0 \mod 2$, *portanto*

$$\frac{n^2-1}{8} \equiv \frac{k^2-1}{8} + \frac{w^2-1}{8} \mod 2$$

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

Então válido para qualquer composto, concluindo a indução.

Agora, para provar o item 1 vamos iniciar com n e m primos ímpares. Sendo

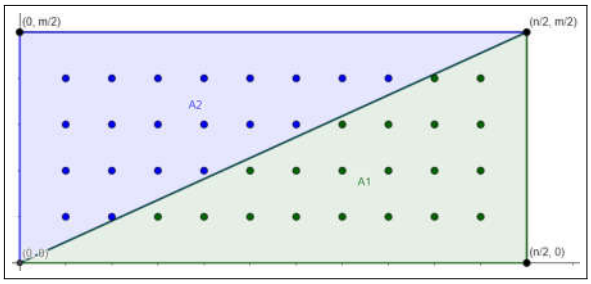
$$A = \{(x,y) \mid x,y \in \mathbb{Z}, 1 \leq x \leq \frac{n-1}{2}, 1 \leq y \leq \frac{m-1}{2}\}$$

$$A_1 = \{(x,y) \in A \mid yn < xm\} \text{ e } A_2 = \{(x,y) \in A \mid yn > xm\}$$

Notemos que $yn = xm$ não possui solução inteira, pois caso contrário teríamos $n \mid x$, mas $0 < x < n$. Assim A_1 e A_2 dividem A em dois subconjuntos disjuntos e obtemos

$$|A_1| + |A_2| = |A| = \frac{n-1}{2}\frac{m-1}{2}$$

Figura 1 – Exemplo da divisão dos pontos em A1 e A2



Fonte: Elaborado pelo autor

Agora para um y fixo o número de pares ordenados $(x,y) \in A_1$ é $\lfloor \frac{xm}{n} \rfloor$, consequentemente

$$|A_1| = \Sigma_{x=1}^{\frac{n-1}{2}} \left\lfloor \frac{xm}{n} \right\rfloor \text{ e de modo análogo } |A_2| = \Sigma_{y=1}^{\frac{m-1}{2}} \left\lfloor \frac{yn}{m} \right\rfloor.$$

Então

$$|A_1|+|A_2| = \frac{n-1}{2}\frac{m-1}{2} = \Sigma_{x=1}^{\frac{n-1}{2}}\left\lfloor \frac{xm}{n} \right\rfloor + \Sigma_{y=1}^{\frac{m-1}{2}}\left\lfloor \frac{yn}{m} \right\rfloor$$

Agora precisamos provar que $|A_1| \equiv \lambda_n(m) \mod 2$ *e de mesmo modo que* $|A_2| \equiv \lambda_m(n) \mod 2$. *Sendo* $xm = n\left\lfloor \frac{xm}{n} \right\rfloor + r_x$, *onde* r_x *representa o resto da divisão de xm por n. Então*

$$m\Sigma_{x=1}^{\frac{n-1}{2}}x = n\Sigma_{x=1}^{\frac{n-1}{2}}\left\lfloor \frac{xm}{n} \right\rfloor + \Sigma_{r_x<\frac{n}{2}}m_x + \Sigma_{r_x>\frac{n}{2}}(n-m_x),$$

ou seja, se $r_x < \frac{n}{2}$ *teremos* $r_x = m_x$, *agora se* $r_x > \frac{n}{2}$ *teremos* $r_x = n - m_x$.

Como n e m são ímpares, módulo 2 teremos

$$n+1 \equiv 2m_x \mod 2 \Rightarrow n - m_x \equiv m_x - 1 \mod 2,$$
$$\Sigma_{x=1}^{\frac{n-1}{2}}x \equiv \Sigma_{x=1}^{\frac{n-1}{2}}\left\lfloor \frac{xm}{n} \right\rfloor + \Sigma_{r_x<\frac{n}{2}}m_x + \Sigma_{r_x>\frac{n}{2}}(m_x-1) \mod 2.$$

Como $\{m_1, m_2, ..., m_{\frac{n-1}{2}}\} = \{1, 2, ..., \frac{n-1}{2}\}$ *temos*

$$\Sigma_{x=1}^{\frac{n-1}{2}}x \equiv \Sigma_{x=1}^{\frac{n-1}{2}}\left\lfloor \frac{xm}{n} \right\rfloor + \Sigma_{x=1}^{\frac{n-1}{2}}x - \Sigma_{r_x>\frac{n}{2}}(1) \mod 2$$

Por definição temos que $\lambda_n(m) = \Sigma_{r_x>\frac{n}{2}}(1)$ *então*

$$\Sigma_{x=1}^{\frac{n-1}{2}}\left\lfloor \frac{xm}{n} \right\rfloor \equiv \Sigma_{r_x>\frac{n}{2}}(1) \mod 2 \Leftrightarrow |A_1| \equiv \lambda_n(m) \mod 2$$

De modo análogo temos que $|A_2| \equiv \lambda_m(n) \mod 2$.

Então pelo lema 2.4.1 concluímos

$$\left(\frac{n}{m}\right)\left(\frac{m}{n}\right) = (-1)^{\lambda_n(m)+\lambda_m(n)} = (-1)^{\frac{n-1}{2}\frac{m-1}{2}}.$$

Agora vamos provar por indução que é válido para qualquer n e m ímpares e primos entre si. Supondo válido para k, w, m, *onde* $n = k.w$, *então pelas propriedades do símbolo de Legendre e Jacobi 2.4.3 temos*

$$\left(\frac{m}{k}\right)\left(\frac{k}{m}\right) = (-1)^{\frac{k-1}{2}\frac{m-1}{2}} \; e \; \left(\frac{m}{w}\right)\left(\frac{w}{m}\right) = (-1)^{\frac{w-1}{2}\frac{m-1}{2}}$$
$$\left(\frac{m}{n}\right)\left(\frac{n}{m}\right) = \left(\frac{m}{k}\right)\left(\frac{m}{w}\right)\left(\frac{k}{m}\right)\left(\frac{w}{m}\right) = (-1)^{\frac{k-1}{2}\frac{m-1}{2}+\frac{w-1}{2}\frac{m-1}{2}} = \left((-1)^{\frac{k-1}{2}+\frac{w-1}{2}}\right)^{\frac{m-1}{2}}$$

Sabemos que $\frac{(k-1)(w-1)}{2}$ *é um número par, então*

$$\frac{n-1}{2} = \frac{(k-1)(w-1)+k+w-2}{2} \equiv \frac{k-1}{2} + \frac{w-1}{2} \mod 2$$

o que encerra a prova.

□

2.5 GRUPO ABELIANOS E TEOREMA DE LAGRANGE

Definição 2.5.1 (Grupo) *Um conjunto G com uma operação $*$ é um grupo se:*

1. *Associatividade: dados $a,b,c \in G$ temos que*

$$a*(b*c) = (a*b)*c$$

2. *Elemento neutro: existe um elemento $e \in G$ tal que, $\forall a \in G$, temos*

$$a*e = e*a = a$$

3. *Elemento inverso: dado um elemento $a \in G$, existe um elemento $b \in G$, o inverso de a, tal que*

$$a*b = b*a = e$$

Denotaremos o elemento b por a^{-1}.

Um grupo finito é um grupo que possui um quantitativo finito de elementos.

Definição 2.5.2 (Grupo Abeliano) *H é um grupo que possui operação comutativa, ou seja, $a*b = b*a$ para quaisquer elementos $a,b \in G$, portanto abeliano.*

$U(n)$ é o grupo abeliano dos elementos invertíveis de $\mathbb{Z}_n$, então

$$U(n) = \{\overline{a} \in \mathbb{Z}_n \mid (a,n) = 1\}.$$

Definição 2.5.3 (Ordem) *O número de elementos de um grupo é a sua ordem.*

Exemplo:

$$|\mathbb{Z}_n| = n$$

Definição 2.5.4 (Subgrupo) *Seja G um grupo com a operação $*$. Um subconjunto não vazio H de G é um subgrupo de G se:*

1. *Para todo $a,b \in H$ temos que $a*b \in H$.*
2. *O elemento neutro de G está em H.*
3. *Para todo $a \in H$, seu inverso a^{-1} também está em H.*

Um exemplo importante de subgrupo é o subgrupo H de G formado pelas potencias de a, onde, sendo e o elemento neutro, $a^0 = e$ e k o menor natural tal que $a^k = e$. Ou seja,

$$H = \{e, a, a^2, ..., a^{k-1}\} = \{a^m | m \in \mathbb{Z}\}$$

denotado por $\langle a \rangle$ e tem ordem k. Um subgrupo que admite um único elemento gerador é chamado de cíclico.

Tomando como base a relação de equivalência

$$y \equiv x \mod H \Leftrightarrow y * x^{-1} \equiv e \mod H \Leftrightarrow y * x^{-1} \in H$$

definimos o que se segue.

Definição 2.5.5 (Classe de Equivalência) *Para um grupo G, de operação $*$, e subgrupo H temos que a classe de equivalência $\hat{x}$ de um elemento $x \in G$ é definida por*

$$\hat{x} = \{y \in G \mid y \equiv x \mod H\}$$

equivale a dizer que

$$\hat{x} = \{h * x \mid h \in H\}$$

Vejamos um exemplo, no grupo $U(10) = \{\overline{1}, \overline{3}, \overline{7}, \overline{9}\}$, considerando o subgrupo $H = \{\overline{1}, \overline{9}\} = \langle \overline{9} \rangle$. Então

$$\hat{1} = \{\overline{1}, \overline{9}\} = H \text{ e } \hat{7} = \{\overline{3}, \overline{7}\}$$

A partir desse momento todos os grupos estudados são abelianos.

G/H denota o conjunto das classes $\hat{x}$ de equivalência módulo H, ou seja,

$$G/H = \{\hat{x} | x \in G\}.$$

Definição 2.5.6 (Grupo Quociente) *Consideremos a seguinte operação em G/H:*

$$\hat{x} * \hat{y} = \widehat{x * y}$$

Verifica-se que a operação $*$ está bem definido, pois o grupo em questão é abeliano. Forma-se G/H, chamado o grupo quociente de G por H.

Para facilitar o entendimento, retomemos o exemplo anterior e então teremos

$$G/H = \{\hat{1}, \hat{7}\}$$

pois $G = \hat{1} \bigcup \hat{7} = \{\overline{1}, \overline{9}\} \bigcup \{\overline{3}, \overline{7}\}$.

Teorema 2.5.1 (Lagrange) *Em um grupo finito, a ordem de qualquer subgrupo divide a ordem do grupo, mais precisamente* $|G| = |H|.|G/H|$.

Demonstração 2.5.1 *Seja G um grupo abeliano finito de operação* $*$ *e H subgrupo de G, temos que*

- *G é união das classes de equivalência distintas relativamente à congruência módulo H, ou seja,* $G = C_1 \bigcup ... \bigcup C_k$.
- *Duas classes de equivalência diferentes têm que ser disjuntas, ou seja,* $C_i \bigcap C_j = \phi$ *quando* $i \neq j$.

Mas isso implica que

$$|G| = |C_1| + ... + |C_k|.$$

Sendo e o elemento neutro temos que $\hat{e} = H$, *de maneira que H coincide com uma das classes de equivalência em* G/H. *Se* $a \notin H$, *então*

$$\hat{a} = \{h * a \mid h \in H\}.$$

Sabemos que â não pode ter ordem maior que H, mas se $|\hat{a}| < |H|$, *devemos ter* $a * h_1 = a * h_2$, *para dois elementos distintos de H. Entretanto, multiplicando pelo inverso de a em G obtemos* $h_1 = h_2$, *logo* $|\hat{a}| = |H|$.

Então $|C_1| = ... = |C_k| = |H| \Rightarrow |G| = k|H|$, *concluindo a prova.*

□

Teorema 2.5.2 *Todo grupo G de ordem prima p é cíclico e qualquer elemento* $a \in G$, *diferente de e, é um gerador de G.*

Demonstração 2.5.2 *Considerando um subgrupo cíclico* $\langle a \rangle$, *aplicando o teorema de Lagrange 2.5.1 temos que a ordem de* $\langle a \rangle$ *divide a ordem de G. Então ordem de* $\langle a \rangle$ *é* 1 *ou p, mas como* $e, a \in \langle a \rangle$, *portanto* $|\langle a \rangle| = p$.

□

Lema 2.5.1 *Sendo G um grupo finito munido de uma operação* $*$ *e* $a \in G$. *Um inteiro positivo t satisfaz* $a^t = e$ *se, e somente se, t é divisível pela ordem de a.*

Demonstração 2.5.3 *Chamaremos de s a ordem de a. Se s divide t então $t = s.r$, para algum inteiro positivo r, e*

$$a^t = (a^s)^r = e.$$

A recíproca é menos imediata. Suponhamos que $a^t = e$. Como a ordem é o menor inteiro positivo s tal que $a^s = e$, então $s \leq t$. Dividindo t por s, temos

$$t = s.q + r \text{ onde } 0 \leq r < s.$$

Assim,

$$e = a^t = (a^s)^q.a^r = a^r,$$

já que $a^s = e$. Como $r < s$ isto só pode acontecer se $r = 0$, ou teríamos uma contradição com o fato de s ser a ordem de a.

□

2.6 ANÉIS, IDEAIS E POLINÔMIOS

Definição 2.6.1 (Anel) *Um conjunto não vazio D é um anel comutativo com unidade se nele temos definidas as operações de adição e multiplicação, satisfazendo as condições:*

- *Para qualquer $a, b, c \in D$*

$$a + (b + c) = (a + b) + c$$
$$a + b = b + a$$
$$a(bc) = (ab)c$$
$$ab = ba$$
$$a(b + c) = ab + ac$$

- $0, 1 \in D$, *tais que*

$$a + 0 = a$$
$$a1 = a$$

- *Para todo $a \in D$ existe seu inverso aditivo b tal que*

$$a + b = 0$$

Denotaremos o elemento b por $-a$.

Definição 2.6.2 (Corpo) *Se H é um anel comutativo com unidade em que todo elemento não nulo a possui inverso multiplicativo a^{-1}, isto é*

$$aa^{-1} = 1$$

então dizemos que H é um corpo.

Um exemplo muito útil de corpo para nosso estudo é o corpo $\mathbb{Z}_n$ para um n primo. Supondo $n = ab$, para $1 < a,b < n-1$, então $\overline{a}\overline{b} = \overline{0}$, portanto para n composto temos que $\mathbb{Z}_n$ é anel, mas não é corpo.

No que segue, consideraremos polinômios f da variável x, com coeficientes em D, expresso da forma

$$f(x) = d_n x^n + ... + d_2 x^2 + d_1 x + d_0$$

onde $n \in \mathbb{N}$ e $d_0, d_1, ..., d_n \in D$. O conjunto de todos os polinômios na variável x, com coeficientes no anel D, será denotado por $D[x]$, que também é um anel com adição e multiplicação usuais de polinômios.

Definição 2.6.3 (Ideal) *Um subgrupo não vazio E do grupo aditivo do anel D é um ideal de D se*

$$\forall\, d \in D \; e \; x \in E \Rightarrow dx \in E$$

Se D um corpo, supondo que E um ideal não nulo de D. Então, E possui um elemento $d \neq 0$. Como D é um corpo, $d^{-1} \in D$. Da definição de Ideal 2.6.3 temos que

$$dd^{-1} = 1 \in E \Rightarrow \forall\, b \in D \mid b.1 = b \in E$$

então $E = D$.

Sendo E um ideal de D gerado por $d_1, ..., d_n$, denotamos por $(d_1, ..., d_n)$.

$$E = (d_1, ..., d_n) = \{d_1 x_1 + ... + d_n x_n \mid x_1, ..., x_n \in D\}$$

O ideal gerado por $d_1, ..., d_n$ será denotado por $(d_1, ..., d_n)$ para diferenciá-lo do subgrupo $\langle d_1, ..., d_n \rangle$ gerado pelos mesmos elementos. Um caso importante é o ideal principal que é um ideal gerado por apenas um elemento, ou seja, $E = (d_1)$. Ele desempenha na teoria dos anéis um papel semelhante ao dos grupos cíclicos na teoria de grupos.

Para deixar clara a diferença entre ideais e subgrupos aditivos de um anel, vamos dar dois exemplos no anel $\mathbb{Z}_n[x]$. O subgrupo aditivo de $\mathbb{Z}_n[x]$ gerado pelo polinômio x é

$$\langle x \rangle = \{\overline{0}, x, \overline{2}x, ..., \overline{n-1}x\}.$$

Contudo, o ideal gerado por x contém todos os polinômios da forma $xp(x)$ onde $p(x)$ é um polinômio qualquer de $\mathbb{Z}_n[x]$. Mas esses são exatamente os polinômios $p(x)$ tais que $p(0) = 0$, então o ideal é

$$(x) = \{p(x) \in \mathbb{Z}_n[x] | p(0) = 0\}.$$

Teorema 2.6.1 *Todo ideal do anel de polinômios em uma variável sobre um corpo é principal.*

Demonstração 2.6.1 *Seja D um corpo e seja E um ideal de $D[x]$. Tanto $\{0\}$ como $D[x]$ são gerados por um elemento, de modo que podemos supor que $(0) \subset E \subset D[x]$.*

Seja g um polinômio não nulo qualquer de E cujo o grau é o menor possível. Para provar que cada elemento de E é divisível por g, tomemos $f \in E$ tal que

$$f = gq + r \text{ onde } r = 0 \text{ ou } grau(r) < grau(g)$$

Como $f, g \in E$, temos $f - gq \in E$, então se $r \neq 0$ temos que $grau(r) < grau(g)$ e $r \in E$. Absurdo, então $r = 0$.

□

Definição 2.6.4 (Polinômio Mônico) *O polinômio $f(x)$ é mônico quando possui o coeficiente líder igual a 1, ou seja, $d_n = 1$*

$$f(x) = x^n + ... + d_2x^2 + d_1x + d_0$$

De volta aos quocientes, digamos que D seja um anel e E seja um ideal de D. Como E é um subgrupo do grupo aditivo de D, podemos construir o grupo quociente D/E, que estará automaticamente dotado de uma adição. De fato, se $e, d \in D$, então denotando por $\hat{e}$ e $\hat{d}$ as classes correspondentes em D/E, teremos que

$$\hat{e} + \hat{d} = \widehat{e+d}.$$

Toda a nossa discussão anterior leva a crer que a multiplicação deva ser definida em D/E por

$$\hat{e}.\hat{d} = \widehat{ed}.$$

Para verificar que esta formula nos dá uma multiplicação bem definida, vamos supor que $e' \in \hat{e}$ e que $d' \in \hat{d}$ são outros representantes desta mesma classe. Isto significa que

$$e' = e + s \text{ e } d' = d + t, \text{ onde } s,t \in E.$$

Multiplicando e' por d' obtemos

$$e'd' = (e+s)(d+t) = ed + (ds + et + st).$$

Mas E é um ideal de D, de modo que $ds + et + st \in E$. Portanto, $\widehat{e'd'} = \widehat{ed}$, o que comprova que a multiplicação está bem definida, então D/E um anel.

Definição 2.6.5 (Anel Quociente) *Consideremos as seguintes operações em D/E:*

$$\hat{e} + \hat{d} = \widehat{e+d}$$

$$\hat{e}.\hat{d} = \widehat{ed}$$

Adição e multiplicação são bem definidas e formam D/E, chamado o anel quociente de D por E.

Pelo teorema 2.6.1 todo ideal E de $D[x]$ é principal gerado por um polinômio $g \in D[x]$. Vamos efetuar o que se segue

- Caracterizar as classes de equivalência de $D[x]/(g)$
- Determinar uma fórmula geral para a multiplicação de classes

Para achar uma fórmula geral para as classes, usaremos o algoritmo da divisão de $D[x]$. Seja $f \in D[x]$, se f tem grau maior que n, podemos dividí-lo por g, obtendo

$$f = qg + r \text{ onde } r = 0 \text{ ou } |r| < n$$

Como $f - r = qg \in (g)$, concluímos que $\hat{f} = \hat{r}$ em $D[x]$. Então mostramos que toda classe de $D[x]/(g)$ tem um representante cujo o grau é menor que n. Além disso, a unicidade do resto da divisão de polinômios mostra que tal representante é único se for mônico.

Como os elementos de $D[x]$ podem ser escritos como combinação linear das potências de x, então os elementos $D[x]/(g)$ podem ser escritos como combinação das potências de $\hat{x}$, que é a classe de x módulo (g). Portanto, pela propriedade distributiva, basta saber multiplicar potências de $\hat{x}$ para saber multiplicar duas classes.

Exemplificando, para $D = \mathbb{Z}_2$ e $g(x) = x^2 - 1$. Nesse caso o anel quociente

$$\mathbb{Z}_2[x]/(x^2-1) = \{\hat{0}, \hat{1}, \hat{x}, \hat{x}+\hat{1}\}$$

então $\mathbb{Z}_2[x]/(x^2-1)$ é um corpo finito com 4 elementos.

Sendo $f(x) = x^n + a_{n-1}x^{n-1} + ... + a_1x + a_0$ o anel quociente $\mathbb{Z}_k[x]/f(x)$ terá k^n elementos, pois cada elemento será da forma

$$\hat{r}_{n-1}\hat{x}^{n-1} + ... + \hat{r}_1\hat{x} + \hat{r}_0.$$

onde, para cada $\hat{r}_i$ teremos k possíveis valores, totalizando k^n combinações.

2.6.1 Fatoração Única de Polinômios

Definição 2.6.6 (Polinômio Irredutível) *É um polinômio, de grau natural, que não pode ser fatorado em polinômios de graus menores.*

Lema 2.6.1 *Sendo D um corpo e $D[x]$ o conjunto de todos os polinômios na variável x. Sejam p, f, g polinômios pertencentes a $D[x]$ e supondo p irredutível em $D[x]$.*

1. *Se p não divide f, então existem polinômios $a, b \in D[x]$ tais que $ap + bf = 1$.*
2. *Se p divide fg, então $p \mid f$ ou $p \mid g$.*

Demonstração 2.6.2 *Para provar a parte (1) sabemos que $D[x]$ é um anel de polinômios em uma única variável, então podemos afirmar, pelo teorema 2.6.1, que qualquer ideal de $D[x]$ será principal. Portanto o ideal de $D[x]$ gerado por f e p é principal, logo existe um polinômio h que gera (f, p). Então h divide f e p, mas como p é irredutível e $p \nmid f$ então h é um polinômio constante. Utilizando as propriedades do corpo temos que o inverso multiplicativo de h, h^{-1} existe, então $h.h^{-1} = 1 \in (f, p)$.*

Para provar a parte (2), se $p \mid f$, então o resultado está provado. Suponhamos $p \nmid f$, então por 1 temos que $ap + bf = 1$, multiplicando por g temos $apg + bfg = g$. Como $p \mid apg$ e pela hipótese 2 temos que $p \mid bfg$, então $p \mid g$.

□

Teorema 2.6.2 (Fatoração Única) *Sejam D corpo e $f \in D[x]$ um polinômio não constante. Então*

$$f = cp_1^{\alpha_1}...p_k^{\alpha_k}$$

onde c é uma constante diferente de zero, $p_1, ..., p_k$ são polinômios mônicos irredutíveis distintos e $\alpha_1, ..., \alpha_k \in \mathbb{N}$. Essa fatoração é única, podendo variar apenas a ordenação dos polinômios.

Demonstração 2.6.3 *Vamos iniciar a prova utilizando indução sobre* $\deg(f)$ *para provar a existência da fatoração.*

Um polinômio de grau 1 é irredutível. Suponha que todo polinômio de grau menor que n admite uma fatoração conforme o teorema. Sendo $\deg(f) = n$*, se irredutível, já está fatorado, caso contrário podemos tomar* $f = d_1 d_2$*, onde* $1 \leq \deg(d_1) < n$ *e* $1 \leq \deg(d_2) < n$*. Então podemos fatorá-los conforme teorema, efetuando a multiplicação dos polinômios encontramos o que desejamos, sendo assim está provada a existência por indução.*

Agora vamos provar a unicidade da fatoração por contradição. Supondo que existem duas maneiras de fatorar f*, ou seja,*

$$f = c p_1^{\alpha_1} ... p_k^{\alpha_k} = c' g_1^{\beta_1} ... g_k^{\beta_k}.$$

Como todo os polinômios são mônicos temos que $c = c'$*. Como são polinômios irredutíveis e pela igualdade temos que* p_1 *divide um certo* g_i*, portanto* $p_1 = g_i$*. Como os polinômios de mesma fatoração são primos entre si, podemos afirmar também que* $\alpha_1 = \beta_i$*. Analogamente para os demais, então provado a unicidade.*

□

Proposição 2.6.3 *Um polinômio de grau n, com coeficientes em um corpo D, não pode ter mais do que n raízes em D.*

Demonstração 2.6.4 $d \in D$ *é uma raiz de* f *se, e somente se,* $x - d \mid f$*. Então* $x - d$ *é um polinômio irredutível de* f*. Contudo, pelo teorema 2.6.2, a soma dos graus dos fatores irredutíveis de* f *não pode exceder o grau de* f*. Portanto,* f *não pode ter mais fatores lineares que seu grau.*

□

Proposição 2.6.4 *Seja D um corpo e* $p \in D[x]$ *um polinômio irredutível. Então o anel quociente* $D[x]/(p)$ *é um corpo. Se D é um corpo finito, então* $D[x]/(p)$ *também será.*

Demonstração 2.6.5 *Vamos provar que toda classe não nula do anel quociente* $D[x]/(p)$ *é invertível. Sabemos que cada uma destas classes admite como representante um polinômio* f *de grau não negativo e menor do que o grau de p,* $\deg(p)$*. Como toda constante não nula é invertível em* $D[x]/(p)$*, podemos supor que* f *tem grau positivo. Então*

$$\hat{f} \in D[x]/(p), \text{ onde } 0 < \deg(f) < \deg(p)$$

Pelo lema 2.6.1 existem $a, b \in D[x]$ *tais que*

$$ap + bf = 1.$$

Como $\hat{p} = \hat{0}$*, temos*

$$\hat{a}\hat{p} + \hat{b}\hat{f} = \hat{1} \Rightarrow \hat{b}\hat{f} = \hat{1}.$$

Portanto $\hat{b}$ *é o inverso de* $\hat{f}$ *em* $D[x]/(p)$*.*

A afirmação final vem do fato de que se D é finito, então existe apenas uma quantidade finita de polinômios em $D[x]$ *com o grau menor do que o grau de p.*

□

2.7 CÁLCULOS ÚTEIS EM $\mathbb{Z}_n[x]$

Os cálculos a seguir serão efetuados em $\mathbb{Z}_n[x]$, o anel de polinômios com coeficientes em $\mathbb{Z}_n$.

Teorema 2.7.1 *Sejam* $n, a \in \mathbb{Z}^+$ *primos entre si. Então*

$$(x + \overline{a})^n = x^n + \overline{a}^n \text{ em } \mathbb{Z}_n[x]$$

se, e somente se, n for primo.

Demonstração 2.7.1 *Temos, usando o binômio de Newton 2.0.1, que*

$$(x + \overline{a})^n = x^n + \overline{a}^n + \Sigma_{k=1}^{n-1} \binom{n}{k} x^k \overline{a}^{n-k}.$$

Então podemos reduzir a prova do teorema a provar que

$$\Sigma_{k=1}^{n-1} \binom{n}{k} x^k \overline{a}^{n-k} = \overline{0} \text{ em } \mathbb{Z}_n[x]$$

se, e somente se, n for primo. Entretanto, como $(a, n) = 1$*, podemos afirmar que*

$$\binom{n}{k} a^{n-k} \equiv 0 \mod n \text{ se, e somente se, } \binom{n}{k} \equiv 0 \mod n.$$

Pelo lema 2.1.1 concluímos a prova.

□

Proposição 2.7.2 *Sejam $n, a \in \mathbb{Z}^+$ primos entre si. Então*

$$(x+\overline{a})^n = x^n + \overline{a} \text{ em } \mathbb{Z}_n[x]$$

se, e somente se, n for primo.

Demonstração 2.7.2 *Pelo teorema 2.7.1 temos que $(x+\overline{a})^n = x^n + \overline{a}^n$ em $\mathbb{Z}_n[x]$ só acorre se, e somente se, n for primo. Note que para n primo, pelo Pequeno Teorema de Fermat 2.2.1, temos*

$$a^n \equiv a \mod n, \text{ por tanto, } \overline{a}^n = \overline{a} \text{ em } \mathbb{Z}_n[x].$$

Por outro lado, se n for composto, a troca não afeta o fato de haver números binomiais que não se anulam módulo n.

□

Proposição 2.7.3 *Sendo a, k e p inteiros positivos e p primo. Então*

$$(x+\overline{a})^{p^k} = x^{p^k} + \overline{a} \text{ em } \mathbb{Z}_p[x].$$

Demonstração 2.7.3 *Vamos efetuar indução em k. Para $k = 0$ é imediato. Se $k = 1$ o resultado equivale à proposição 2.7.2.*

Supondo que para algum k seja válido

$$(x+\overline{a})^{p^k} = x^{p^k} + \overline{a} \text{ em } \mathbb{Z}_p[x],$$

então

$$(x+\overline{a})^{p^{k+1}} = ((x+\overline{a})^{p^k})^p = (x^{p^k} + \overline{a})^p \text{ em } \mathbb{Z}_p[x].$$

Pela hipótese de indução temos

$$(x+\overline{a})^p = x^p + \overline{a} \text{ em } \mathbb{Z}_p[x].$$

Sem perda de generalização podemos substituir x por x^{p^k}, então

$$(x^{p^k} + \overline{a})^p = x^{p^{k+1}} + \overline{a} \text{ em } \mathbb{Z}_p[x]$$

concluindo a demonstração.

□

3 RIVEST SHAMIR ADLEMAN

Tomando como base o estudo feito no livro (ROUSSEAU *et al.*, 2008) que fala sobre várias aplicações da matemática na tecnologia vamos explicar de modo detalhado o algoritmo Rivest Shamir Adleman.

Esse é um dos algoritmos mais seguros de encriptação da atualidade, mais conhecido como algoritmo RSA, desenvolvido por Ronald Rivest, Adi Shamir e Leonard Adleman em 1978, três matemáticos geniais que mudaram a história da criptografia. O princípio do algoritmo é construir chaves públicas e privadas utilizando números primos. Uma chave é uma informação restrita que controla toda a operação dos algoritmos de criptografia. No processo de codificação uma chave é quem dita a transformação do texto puro (original) em um texto criptografado. A chave privada é uma informação pessoal que permanece em posse da pessoa - não publicável. A chave pública é uma informação, associada a uma pessoa, que é distribuída a todos.

No funcionamento deste sistema criptográfico é necessário basicamente fatorar inteiros grandes para quebrá-lo. É uma operação conceitualmente simples de ser efetuada, dividir um inteiro composto em um produto de seus fatores primos. Mas algumas propriedades dos computadores permitem que esse algoritmo seja eficiente, são elas: sua dificuldade de fatorar números grandes; facilidade de construir e testar números primos grandes.

É iniciado a configuração do sistema de criptografia pelo computador do receptor e as etapas seguem abaixo:

1. O receptor escolhe dois primos grandes p e q, normalmente com cem dígitos cada, e calcula $n = pq$. Desse modo o número n, que será a *chave pública*, terá aproximadamente duzentos dígitos. Assim, mesmo conhecendo n, um computador não consegue fatorá-lo para encontrar p e q em tempo hábil.
2. O receptor calcula $\varphi(n)$, onde φ é definida como a função de Euler 2.2.1. Note que é necessário conhecer p e q. Assim, calcular $\varphi(n)$ sem conhecer a fatoração de n parece ser tão difícil quanto a fatoração de n.
3. O receptor escolhe um número $e \in \{1,...,n-1\}$ relativamente primo para $\varphi(n)$. O número e é a *chave de criptografia*. Esse número é público e é usado pelo remetente para codificar a mensagem seguindo as instruções disponibilizadas publicamente pelo destinatário.
4. Existe um número $d \in \{1,...,n-1\}$ tal que $ed \equiv 1 \mod \varphi(n)$. O inteiro d, construído pelo receptor, é a chave de descriptografia. Essa é a *chave privada*,

permanece secreta e permite que o receptor decodifique suas mensagens recebidas.

5. O remetente, já conhecendo n e e, deseja enviar uma mensagem que consiste em um inteiro $\omega \in \{1,...,n-1\}$. Dentre o conjunto de 1 até $n-1$ temos apenas p e q não relativamente primos com n. Como n é um número com duzentos dígitos em média, temos que a probabilidade de que $(\omega,n) = 1$ é muito próximo de 100%, portanto ω é considerado relativamente primo com n. Para codificá-lo, o remetente calcula o resto r da divisão de ω^e por n. Assim, temos que $\omega^e \equiv r \mod n$, com $r \in \{1,...,n-1\}$. O inteiro calculado r é a mensagem criptografada. O remetente envia r. Como veremos mais adiante, é fácil para um computador calcular r, mesmo quando ω, e e n são muito grandes.
6. O receptor recebe uma mensagem criptografada r. Para decifrar esta mensagem, o receptor calcula $r^d \mod n$. que sempre produzirá precisamente a mensagem inicial ω.

Efetuando um exemplo para um par de primos pequenos para auxiliar o entendimento.

Exemplo 3.0.1 *O receptor escolhe os primos $p = 5$, $q = 17$ e executa as etapas que se seguem.*

1. *Calcula a chave pública $n = pq = 85$.*
2. *Calcula $\varphi(85) = (5-1)(17-1) = 4.16 = 64$.*
3. *Escolhe a chave de criptografia $e = 19$, que pertence ao conjunto $\{1,...,84\}$ e é relativamente primo com $\varphi(85)$. Esse número é conhecido por todos e é usado pelo remetente para codificar a mensagem.*
4. *Calcula a chave privada d, onde $ed \equiv \mod \varphi(85)$ utilizando o Algoritmo de Euclides:*

$$\begin{cases} 64 = 19.3+7 \\ 19 = 7.2+5 \\ 7 = 5.1+2 \\ 5 = 2.2+1 \end{cases} \Longrightarrow \begin{cases} 1 = 5-2.2 \\ = 5-2(7-5) = 3.5-2.7 \\ = 3(19-7.2)-2.7 = 3.19-8.7 \\ = 3.19-8(64-3.19) = 27.19-8.64 \end{cases}$$

$$19.27 = 8.64+1 \Longrightarrow 19.27 \equiv 1 \mod 64 \Longrightarrow d = 27$$

5. *Digamos que o remetente deseja enviar a mensagem $\omega = 13$, relativamente primo*

com 85. $13^{19} \equiv r \mod 85$.

$$13^2 = -1 \equiv 41 \mod 85$$
$$13^{18} = (13^2)^9 \equiv (-1)^9 = -1 \mod 85$$
$$13^{19} = 13^{18}.13 \equiv -13 \equiv 72 \mod 85$$

Então $r = 72$.

6. *O receptor recebe uma mensagem criptografada* 72. *Para decifrar esta mensagem, o receptor, utilizando da chave privada* $d = 27$, *calcula* $72^{27} \equiv \omega \mod 85$

$$72 \equiv -13 \mod 85$$
$$72^2 \equiv (-13)^2 = 13^2 \equiv -1 \mod 85$$
$$72^{26} = (72^2)^{13} \equiv (-1)^{13} = -1 \mod 85$$
$$72^{27} = 72^{26}.72 \equiv (-1)(-13) = 13 \mod 85$$
Mensagem recebida foi $\omega = 13$

A mensagem 13 foi enviado criptografada com a chave pública 19, recebida pelo destinatário como 72. O destinatário utilizando a chave privada 27 descriptografou a mensagem e então encontrou 13. Conhecendo a *chave pública* e a chave de criptografia não tem como interceptar a mensagem e efetuar as operações inversas para decodificá-la, pois estamos trabalhando com congruência.

Agora vamos provar que a mensagem recebida depois de descriptografada é a mesma envia pelo emitente.

Demonstração 3.0.1 (RSA) *A criptografia e descriptografia RSA são inversas uma da outra: se criptografarmos uma mensagem m, onde* $(m,n) = 1$, *com a, onde* $m^e \equiv a \mod n$, *então a descriptografia sempre gerará a mensagem original m, isto é,* $a^d \equiv m \mod n$.

Se $m^e \equiv a \mod n$, *então* $a^d \equiv (m^e)^d = m^{ed} = m^{k\varphi(n)+1} = (m^{\varphi(n)})^k.m \equiv 1^k.m \equiv m \mod n$.

□

4 CUSTO DE UM ALGORITMO

Antes de apresentar os testes de primalidade temos que conversar sobre como medir o custo, ou tempo de execução, de um algoritmo, tomamos como referência o livro (COUTINHO, 2004). Um bom modo para medir o custo de um algoritmo é quantificar as operações aritméticas elementares que são executadas. Operações aritméticas elementares são operações efetuadas sobre os algarismos, podendo ser adição, subtração, multiplicação, divisão ou cálculo do resto. Como vamos implementar os algoritmos em um computador, então trabalharemos com o sistema binário.

Claro que o quantitativo de operações aritméticas elementares a serem executadas depende do tamanho da entrada que é fornecida ao algoritmo.

Começando por observar quantas operações são necessárias para somar dois inteiros. Sendo $a = (a_{k-1}, ..., a_1, a_0)_2$ e $b = (b_{k-1}, ..., b_1, b_0)_2$ a representação binária dos inteiros a e b, onde os $a_i, b_i \in \{0, 1\}$. Portanto

$$a = a_{k-1}2^{k-1} + ... + a_1 2 + a_0$$

$$b = b_{k-1}2^{k-1} + ... + b_1 2 + b_0$$

Para calcular $s = a + b$, onde $s = s_{k-1}2^{k-1} + ... + s_1 2 + s_0$ começamos calculando s_0. Se $a_0 + b_0$ é menor do que 2, então $s_0 = a_0 + b_0$. Se $a_0 + b_0 = 2$, então $s_0 = 0$ e se soma 1 na casa seguinte, esse valor denominamos de reserva. Foi efetuada até aqui uma operação aritmética.

Para calcular $s_1 = a_1 + b_1$ efetuaremos uma ou duas operações, dependendo se a reserva é nula ou não. Se nula teremos uma operação, já se não nula efetuaremos duas operações aritméticas elementares.

É difícil afirmar com exatidão a quantidade de operações aritméticas, pois depende da quantidade de reservas não nulas. Entretanto, é fácil ver que não necessitamos calcular mais do que $2k$ operações aritméticas elementares, neste caso.

Precisamos ficar atentos no fato de que além das operações aritméticas elementares o computador tem que realizar muitas outras operações para efetuar uma soma. A comparação de bits é uma delas que tem como uma de suas funções ver se a reserva é zero. Existem outras, algumas não sofrendo influência do número tratado, como é o caso do acesso à memória. Mesmo sabendo que essas operações afetam no tempo que o computador leva para executar um algoritmo, vamos relevá-las e nos atermos no que cabe a matemática contida.

Agora vamos trabalhar a multiplicação e descobrir quantas operações aritméticas elementares serão necessárias para multiplicar a por b. Vamos multiplicar primeiro b por cada bit de a e depois somar os resultados. Mas, quando multiplicamos b por um bit a_i de a se houver reserva teremos que efetuar um produto e uma soma de bits. Teremos exatamente k produtos elementares e como não pode ter mais do que $k-1$ reservas, então teremos no máximo $k-1$ somas. Portanto, quando multiplicamos b por a_i necessitamos de no máximo de $2k-1$ operações aritméticas elementares. Então, para calcular o produto de b por cada um dos bits de a, teremos que realizar um valor menor ou igual a $k(2k-1)$ operações aritméticas elementares.

Para calcular ab vamos somar estes produtos de b por bits de a, atentando a ordem de grandeza de cada bit, ou seja, se

$$ba_i = (s_{i,k+1}, ..., s_{i,1}, s_{i,0})_2$$

então podemos escrever a soma da forma

$$\begin{array}{ccccccc}
 & & & s_{0,k+1} & \cdots & s_{0,1} & s_{0,0} \\
 & & s_{1,k+1} & \cdots & s_{1,1} & s_{1,0} & \\
 & s_{2,k+1} & \cdots & s_{2,1} & s_{2,0} & & \\
 & \vdots & \vdots & \vdots & & & \\
s_{k,k+1} & \cdots & s_{k,0} & & & &
\end{array}$$

Para somarmos as duas primeiras linhas efetuaremos $k+1$ somas diretas e no máximo $k+1$ somas de reservas, portanto $2(k+1)$ somas para cada soma de linhas. Como temos $k+1$ linhas teremos k somas de linhas, então $2k(k+1)$ operações no máximo para somar as linhas. Como já havíamos efetuado $k(2k-1)$ operações, então totalizamos

$$2k(k+1) + k(2k-1) = 2k^2 + 2k + 2k^2 - k = 4k^2 + k$$

Unificando tudo já exposto, temos que para calcular ab necessitamos realizar no máximo $4k^2 + k$ operações aritméticas elementares.

Notemos que para descobrir o número aproximado de operações aritméticas elementares para calcular ab tivemos que desprezar muito mais detalhes do que para soma. Como o nosso objetivo aqui é apenas ter um meio de quantificar o custo operacional de um algoritmo de modo aproximado para permitir diferenciar um algoritmo exponencial de um polinomial. Por isso, o que mais nos importa é a maior potência de k na expressão.

Vamos utilizar a notação O para formalizar o custo. Sejam f e g duas funções com contradomínio $\mathbb{R}$, e com domínio $\mathbb{N}$ ou $\mathbb{R}$. Tomamos que o custo de $f(x)$ é $O(g(x))$ se

existem constantes C_1 e C_2 tais que $f(x) \leq C_1 g(x)$ toda vez que $x > C_2$. Por exemplo, se $f(x)$ é um polinômio de grau n com coeficientes reais, então o custo de $f(x)$ é $O(x^n)$. Para provar a veracidade dessa afirmação, tomemos $f(x) = a_n x^n + ... + a_0$, onde $a_n, ..., a_0$ pertencem aos reais e $a_n \neq 0$. Escolhemos um real $b > a_n$. Como

$$\lim_{x \to \infty} \frac{f(x)}{bx^n} = \frac{a_n}{b} < 1,$$

então sempre que x for grande o suficiente, teremos $f(x) \leq bx^n$. Isso equivale a dizer que o custo de $f(x)$ é $O(x^n)$, como queríamos demonstrar.

As propriedades que se seguem da notação O são facilmente demonstradas, e usaremos sem maiores esclarecimentos. Suponhamos que f_1, f_2, g_1, g_2 são funções com contradomínio $\mathbb{R}$, e domínio $\mathbb{N}$ (ou $\mathbb{R}$). Se o custo de f_1 é $O(g_1)$ e de f_2 é $O(g_2)$, então

- O custo de $f_1 + f_2$ é $O(max\{g_1, g_2\})$
- O custo de $f_1 f_2$ é $O(g_1 g_2)$

Em particular, quando o custo de $f_1 = f_2$ é $O(g_1)$, então o custo de $f_1 + f_2$ é $O(g_1)$ e de $f_1 f_2$ é $O(g_1^2)$.

Utilizando da notação, a soma de dois números com k bits cada teremos um custo de $O(k)$, já para o produto $O(k^2)$. É fácil ver que para a subtração teremos um custo de $O(k)$; é bem mais complicado mostrar que para a divisão teremos um custo de $O(k^2)$. Tomemos isso como verdade.

Como, para n um inteiro positivo, temos, aproximadamente, $\log_2 n$ bits, podemos concluir que para somar e subtrair dois inteiros menores do que ou iguais a n teremos um custo máximo de $O(\log_2 n)$. Já para o produto e a divisão teremos um custo máximo de $O((log_2 n)^2)$.

Ao longo desse trabalho será bem mais útil calcular em $\mathbb{Z}_n$, e não em $\mathbb{Z}$. Portanto precisamos determinar o custo das operações com classes módulo n. Sendo $a, b \in \mathbb{Z}_n$, definimos a adição e a multiplicação de classes por

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

Iniciamos a operação com os representantes das classes que pertencem ao conjunto $\{0, 1, ..., n-1\}$, depois o resultado é reduzido módulo n. Tomando $0 \leq a, b \leq n-1$, podemos calcular $a+b$ ao custo $O(log_2 n)$ e ab ao custo $O((log_2 n)^2)$. Para determinar o resíduo módulo n de $a+b$ e de ab, é basicamente dividi-los por n e a classe correspondente será o resto. Como $a+b \leq 2n$ e $ab \leq n^2$, a divisão por n terá um custo de $O((log_2 n)^2)$. Sendo assim, o custo total, tanto da adição quanto do produto em $\mathbb{Z}_n$, será $O((log_2 n)^2)$.

Agora vamos encontrar um divisor primo de $n \in \mathbb{N}$. De partida, precisamos determinar as operações aritméticas que o algoritmo efetuará; cada uma destas operações é executada apenas uma vez, a cada laço que o algoritmo completa. Sendo os números somados menores que $\sqrt{n}$, teremos um custo de execução por laço de $O(log_2 n)$. Já para a divisão teremos um custo de $O((log_2 n)^2)$, pois o dividendo sempre será n. Assim, cada laço terá um custo da ordem de $O((log_2 n)^2)$.

Nos resta estimar o número de laços que serão executados, Vamos trabalhar com o pior caso, onde n é primo. Portanto, o algoritmo executa $\lfloor\sqrt{n}\rfloor$ laços com um custo de $O((log_2 n)^2)$ para cada, dando um custo total da ordem de $O(\lfloor\sqrt{n}\rfloor(log_2 n)^2)$. Lembrando que, quando $a \in \mathbb{R}$, então $\lfloor a \rfloor$ representa a parte inteira de a.

Ao comparar com os algoritmos analisados anteriormente, vemos que é muito mais custoso. Temos, neste caso, uma função exponencial do número de bits, pois $\sqrt{n} = 2^{\frac{\log_2 n}{2}}$. Por conta disso, o número de operações cresce rapidamente quando o número de bits da entrada aumenta, tornando o algoritmo muito lento.

Formalizando a diferença entre um algoritmo eficiente e um que não é eficiente, da seguinte maneira:

Continuamos medindo o custo de um algoritmo em função do número de bits de n, que é aproximadamente igual a $\log_2 n$. Sendo X um algoritmo e $n \in \mathbb{N}$ entrada de X. Teremos X como um algoritmo polinomial se o número de operações aritméticas elementares executados for, para um $r \in \mathbb{N}$, da ordem de

$$O((log_2 n)^r)$$

Teremos X como um algoritmo exponencial se o número de operações aritméticas elementares executados for no mínimo da ordem de

$$O(2^{log_2 n}) = O(n)$$

Então, para o nosso entendimento, um algoritmo é eficiente e rápido quando polinomial e ineficiente e lento quando exponencial.

5 TESTES DE PRIMALIDADE

Antes de partir para os testes de primalidade, devemos ter em mente nosso objetivo: encontrar um número primo grande, testar se um certo número é primo ou apenas ter uma probabilidade alta de o número ser primo.

Existem testes de primalidade determinísticos que em sua maioria têm algoritmos exponenciais; uma exceção é o teste determinístico AKS que possui algoritmo polinomial. Já os testes probabilísticos são em sua maioria mais rápidos e polinomiais, mas não certifica que um número é primo. Tomando como base os livros (GALLIER; QUAINTANCE, 2019), (DIETZFELBINGER, 2004), (COUTINHO, 2004), (RIESEL, 2012) e (MARTINEZ, 2015) escrevemos esse capítulo.

5.1 TESTES PROBABILÍSTICOS

Este tipo de teste é extremamente útil em aplicações como em criptografia, onde é importante criar primos relativamente grandes, mas não existe a preocupação com demonstrações ou com perfeição absoluta, pois com uma probabilidade boa de ser primo já garante que será difícil para o computador descobrir seus divisores e quebrar a criptografia.

5.1.1 Solovay e Strassen

O teste desenvolvido por Robert Martin Solovay e Volker Strassen teve um papel importante, pois foi o primeiro teste a ser popularizado, mostrando a real viabilidade da criptografia de chave pública, em particular o RSA, porém não é mais utilizado devido a criação de testes mais eficientes. O teste de Solovay e Strassen nos permite determinar se um número ímpar é composto ou provavelmente primo usando o Critério de Euler 2.4.2 e o Símbolo de Jacobi 2.4.3.

Pelo Critério de Euler temos para $a \in \{2,...,n-1\}$ e n primo, $\left(\frac{a}{n}\right) \equiv a^{\frac{n-1}{2}} \mod n$. Utilizando esse fato concluir

1. Se n é primo então $\left(\frac{a}{n}\right).a^{\frac{n-1}{2}} \equiv 1 \mod n$.
2. Se $\left(\frac{a}{n}\right).a^{\frac{n-1}{2}} \not\equiv 1 \mod n$ então n é composto.
3. Se n composto temos que a probabilidade de $\left(\frac{a}{n}\right).a^{\frac{n-1}{2}} \equiv 1 \mod n$ é menor que $\frac{1}{2}$.
4. Se para l diferentes valores de a tivermos $\left(\frac{a}{n}\right).a^{\frac{n-1}{2}} \equiv 1 \mod n$, então a probabilidade de n ser composto é menor do que $(\frac{1}{2})^l$, ou seja, a probabilidade de ser

realmente primo é maior do que $1-\frac{1}{2^l}$.

Exemplo 5.1.1 *Para* $n = 325 = 5^2.13$ *temos que* $7^{162} \equiv 324 \equiv -1 \mod 325$ *e* $\left(\frac{7}{325}\right) = -1$*, ou seja,* $\left(\frac{7}{325}\right).7^{\frac{324}{2}} \equiv (-1)^2 \equiv 1 \mod 325$*, mas 325 é composto. Denominamos de pseudo-primo de Euler na base a o número n que mesmo composto atende ao critério de teste , ou seja,* 325 *é um pseudo-primo na base* 7*.*

Agora para a base 2 *temos que* $2^{\frac{325-1}{2}} \equiv 129 \mod 325$*. Como* $\left(\frac{2}{325}\right) \in \{-1,1\}$*, então* $\left(\frac{2}{325}\right).2^{\frac{325-1}{2}} \equiv \pm 129 \not\equiv 1 \mod 325$*. Já na base* 2 *concluímos que* 325 *é composto. Quanto mais base testarmos, mais preciso será nosso resultado.*

Segundo o livro (KATZ *et al.*, 1996), que apresentou o mesmo como fato sem demonstração, se n é um ímpar composto, então ele será *pseudo-primo de Euler* de no máximo $\frac{\varphi(n)}{2}$ bases.

Para cada valor de a tivemos um custo de $O(\log_2^3 n)$.

5.1.2 Miller-Rabin

O teste desenvolvido por Gary Miller surgiu determinístico supondo a veracidade da *Hipótese de Riemann Estendida*, sendo posteriormente modificada por Michael Rabin para um teste probabilístico, conseguindo desse modo eliminar a dependência da *Hipótese de Riemann Estendida*. É hoje em dia, a base para a maioria dos métodos "rápidos" pelos quais os sistemas de computação algébrica certificam que um número é primo. Entretanto o método não garante que um número é primo. Fundamenta-se no pequeno teorema de Fermat 2.2.1 que diz $\forall a \in \mathbb{N}$, tal que $1 < a < n$ e n primo

$$a^{n-1} \equiv 1 \mod n$$

mas nem sempre a recíproca é válida.

Quando n composto e $a^n \equiv a \mod n$ dizemos que n é um *pseudo-primo de Fermat na base a*. Um fato interessante é que existem raros, mas infinitos números compostos n, chamados de *Números de Carmichael*, que são pseudo-primo com todas as bases naturais $1 < a < n$. O menor *Números de Carmichael* é 561. Pomerance provou em sua obra (ALFORD; GRANVILLE; POMERANCE, 1994) que existem infinitos *Números de Carmichael*.

Modificando a ideia de usar a recíproca do pequeno teorema de Fermat para ter um teste de primalidade mais preciso. O que faremos é determinar uma propriedade que todo primo satisfaz e que é mais forte que o pequeno teorema de Fermat.

Tomemos n um ímpar natural. Para efetuar o teste escolhemos um inteiro $1 < b < n-1$, que chamaremos de *base do teste*. Como n é ímpar, $n-1$ é par e podemos escrever $n-1 = 2^k q$, sendo q um inteiro ímpar e $k > 0$, ou seja, encontramos a maior potência de 2 em $n-1$. Calcularmos a seguinte sequência de potências módulo n

$$b^q, b^{2q}, b^{2^2 q}, ..., b^{2^{k-1} q}, b^{2^k q}.$$

Vamos analisar essa sequência quando n for primo. Primeiro, pelo pequeno teorema de Fermat, temos

$$b^{2^k q} \equiv 1 \mod n.$$

Digamos que j é a menor potência tal que $b^{2^j q} \equiv 1 \mod n$. Se $j > 0$ podemos usar produto notável para escrever

$$n | (b^{2^{j-1} q} - 1)(b^{2^{j-1} q} + 1).$$

Como supomos n primo e que $n | b^{2^j q} - 1$, então $n | b^{2^{j-1} q} - 1$ ou $n | b^{2^{j-1} q} + 1$. Como j é o menor expoente tal que $b^{2^j q} - 1$ é divisível por n, portanto $n \nmid b^{2^{j-1} q} - 1$, então $n | b^{2^{j-1} q} + 1$, ou seja, $b^{2^{j-1} q} \equiv -1 \mod n$. Note, entretanto, que estes cálculos dependem do fato de que $j > 0$ e, portanto, se $j = 0 \Rightarrow b^q \equiv 1 \mod n$. Concluímos, assim, que:

Quando n é primo temos que ou a sequência

$$b^q, b^{2q}, b^{2^2 q}, ..., b^{2^{k-1} q}, b^{2^k q}$$

módulo n, começa com 1, ou tem -1 em alguma de suas posições (podendo ser na primeira). Portanto, se nenhum destes dois fatos ocorrerem para n, com respeito a algum $0 < b < n$, então podemos afirmar que n é composto.

Analisando em detalhe o argumento utilizado para chegar à conclusão do parágrafo acima, verificamos que a primalidade de n foi usada em dois pontos cruciais. Em primeiro lugar, precisamos que n fosse primo para garantir a existência de um 1 em alguma posição da sequência. Em segundo lugar, usamos que n é primo para assegurar que, se o produto notável é divisível por n, o mesmo ocorreria com um dos seus fatores.

O problema é que, mais uma vez, temos um teste que apenas detecta se o número é composto. A esse teste denominamos de *teste forte*, pois diferente do teste baseado apenas no pequeno teorema de Fermat esse não possui nada que se assemelhe aos *Números de Carmichael*.

O teste forte foi proposto originalmente por M. Artjuhov em um artigo publicado em 1967. Ganhou notoriedade através dos trabalhos de M. Rabin e de G. L. Miller em 1976. O

que eles propuseram, foram maneiras de tornar o teste mais efetivo, aplicando-o a várias bases diferentes. Para entender melhor como isto é possível, se o teste forte tiver saída inconclusiva quando aplicado a mais de $n/4$ bases diferentes pertencentes ao intervalo que varia de 1 até $n-1$, então n será necessariamente primo.

Para números muito grades é inviável efetuar $n/4$ testes com bases diferentes. Mesmo com essa verdade, há uma maneira muito mais satisfatória de interpretar esta afirmação.

Suponha, como antes, que n é um inteiro ímpar composto. Note que, ao escolher b aleatoriamente, há uma probabilidade de apenas 1/4 de que b seja escolhido de modo que n teste inconclusivo com respeito ao teste forte. Portanto, se um dado n testar inconclusivo para uma base b então a probabilidade de que seja composto é de $1/4$. Entretanto, se testarmos duas bases, e se n testar inconclusivo para ambas, então a probabilidade cai para $(1/4)^2 = 1/16$. Continuando assim vemos que, se n testar inconclusivo para l bases distintas, então a probabilidade de que seja composto caiu para $(1/4)^l$. É claro que um tal número é quase que certamente primo. Nesse fato que se sustenta o teste de primalidade de *Miller-Rabin*. Para cada teste tivemos um custo de $O(\log_2^3 n)$.

Uma maneira de modificar o algoritmo de Miller-Rabin para torná-lo determinístico é testar todos os valores da base a em um intervalo suficientemente grande: essa generalização da hipótese de Riemann implica que o intervalo de 1 até $2(\ln n)^2$ já é grande o bastante. Este algoritmo é rápido e geral, mas infelizmente depende de uma conjectura.

5.2 TESTES DETERMINÍSTICOS

Nesse grupo de testes temos a certeza que o número é primo, mas em sua maioria o tempo de processamento é bem maior do que os que antecedemos.

5.2.1 Crivo de Eratóstenes

O Crivo de Eratóstenes é considerado como o primeiro algoritmo de primalidade inventado, também conhecido como Peneira de Eratóstenes, é atribuído a este matemático grego de Cirene no que hoje é a Líbia que viveu cerca de 276-194 a.C. e ensinou em Alexandria.

Para encontrar todos os números primos menores ou iguais a n, listamos todos os inteiros de 2 a n. Nós então trabalhamos nosso caminho na lista. O primeiro inteiro 2 é primo. Nós cortamos todos os múltiplos de 2 que são maiores que ele. O primeiro inteiro após 2 que não foi riscado é o 3, que é primo. Nós cortamos todos os múltiplos de 3 que são maiores que ele.

Continuamos dessa maneira. Quando nós tivermos encontrado um novo primo, nós cortamos todos os múltiplos desse novo primo que são maiores do que ele, em seguida, passa para o próximo número inteiro que não foi cortado e que será primo.

Uma das coisas que Eratóstenes percebeu foi que nós não temos que continuar isso todo o caminho até n. Uma vez que encontramos um primo maior que a raiz quadrada de n, todos os inteiros restantes que possuem e não foram riscados são primos. Se algum deles fosse composto então eles teriam que ter um fator menor ou igual a sua raiz quadrada.

$$n = ab \Longrightarrow a \leq \sqrt{n} \text{ ou } b \leq \sqrt{n}$$

Com uma complexidade de $O(\sqrt{n}\log_2^2 n)$.

5.2.2 Agrawal, Kayal e Saxena

Em agosto de 2002, Manindra Agrawal, Neeraj Kayal e Nitin Saxena anunciaram um desenvolvimento espetacular, um teste de primalidade determinístico em tempo polinomial, mais conhecido como teste AKS. O novo teste, além de solucionar algo procurado por muitos, o fez de forma bem simples. Esse teste, não diferente de alguns outros citados aqui, se baseia no pequeno teorema de Fermat 2.2.1.

O lema que se segue será usado como base para a demonstração do teorema.

Lema 5.2.1 *Sejam $n, r, a, p \in \mathbb{Z}^+$ com p primo. Suponha que*

$$(\hat{x}+\hat{a})^n = \hat{x}^n + \hat{a} \text{ em } \mathbb{Z}_p[x]/(x^r-1)$$

Então para quaisquer que sejam os inteiros não negativos i, j temos

$$(\hat{x}+\hat{a})^{n^i p^j} \equiv \hat{x}^{n^i p^j} + \hat{a} \text{ em } \mathbb{Z}_p[x]/(x^r-1)$$

Demonstração 5.2.1 *Vamos iniciar efetuando indução em i,*

$$(\hat{x}+\hat{a})^{n^i} \equiv \hat{x}^{n^i} + \hat{a} \text{ em } \mathbb{Z}_p[x]/(x^r-1).$$

É imediato para $i = 0$ e equivale à hipótese $i = 1$.

Supondo a congruência válida para algum i, então existe um $q(x) \in \mathbb{Z}_p[x]$ tal que

$$(x+a)^n = x^n + a + q(x)(x^r-1)$$

Substituindo x por x^{n^i} nesta equação, obtemos

$$(x^{n^i}+a)^n = (x^{n^i})^n + a + q(x^{n^i})((x^{n^i})^r-1).$$

Mas

$$(\hat{x}^{n^i})^r - \hat{1} = (\hat{x}^r - \hat{1} + \hat{1})^{n^i} - \hat{1} = \hat{1} - \hat{1} = \hat{0} \text{ em } \mathbb{Z}_p/(x^r - 1),$$

de modo que

$$(\hat{x}^{n^i} + \hat{a})^n = \hat{x}^{n^{i+1}} + \hat{a} \text{ em } \mathbb{Z}_p/(x^r - 1).$$

Entretanto, pela hipótese de indução,

$$(\hat{x} + \hat{a})^{n^{i+1}} = ((\hat{x} + \hat{a})^{n^i})^n = (\hat{x}^{n^i} + \hat{a})^n \text{ em } \mathbb{Z}_p/(x^r - 1).$$

Portanto,

$$(\hat{x} + \hat{a})^{n^{i+1}} = \hat{x}^{n^{i+1}} + \hat{a} \text{ em } \mathbb{Z}_p/(x^r - 1),$$

o que conclui a prova da indução.

Pela proposição 2.7.3, temos que

$$(\hat{x}^{n^i} + \hat{a})^{p^j} = \hat{x}^{n^i p^j} + \hat{a} \text{ em } \mathbb{Z}_p.$$

Contudo, qualquer igualdade em $\mathbb{Z}_p[x]$ também será em $\mathbb{Z}_p[x]/(x^r - 1)$, concluindo a demonstração.

□

Solucionamos o alto custo de um teste de primalidade baseado em $(x + \overline{a})^n$ em $\mathbb{Z}_n[x]$ substituindo por $(\hat{x} + \hat{b})^n$ em $\mathbb{Z}_n[x]/(x^r - 1)$, para uma quantidade suficientemente grande de primos r. Este último cálculo tem um custo menor que o anterior, pois a potência de x está limitada a x^{r-1} pela congruência.

Teorema 5.2.1 *Sejam $n \geq 2$ e $r \geq 1$ inteiros e seja $S = \{1, 2, ..., r\}$. Suponhamos:*

1. *r é primo e $r \nmid n$.*
2. *$(n, s - s') = 1$ para qualquer par de elementos distintos em S.*
3. *$\binom{2r-2}{r} \geq n^{2d\left\lfloor \sqrt{\frac{r-1}{d}} \right\rfloor}$ vale para qualquer inteiro positivo d que divide $\frac{r-1}{v}$, onde v é a ordem de $\overline{n}$ em $U(r)$.*
4. *$(x + s)^n \equiv x^n + s \mod x^r - 1, n$, para todo $s \in S$.*

Então, n é potência de um primo.

Demonstração 5.2.2 *Se p um primo ímpar, tal que $p|n$, com $(n,r) = 1 \Longrightarrow (p,r) = 1$. Portanto, as classes de $\overline{n}$ e $\overline{p}$ são invertíveis em $\mathbb{Z}_r$. Podemos construir um grupo quociente $U(r)/\langle \overline{n}, \overline{p} \rangle$ tal que, supondo de ordem d. Por r ser primo temos que $|U(r)| = r-1$ e pelo teorema de Lagrange 2.5.1 temos que $|\langle \overline{n}, \overline{p} \rangle| = \frac{r-1}{d}$. A ordem v de $\langle \overline{n} \rangle$ em $U(r)$ divide a ordem de $\langle \overline{n}, \overline{p} \rangle$, portanto $|U(r)/\langle \overline{n}, \overline{p} \rangle|$ divide $|U(r)/\langle \overline{n} \rangle|$, ou seja, d divide $\frac{r-1}{v}$, conforme assumimos na hipótese 3.*

Suponhamos que $m_1, ..., m_d$ são os inteiros que representam as classes que correspondem aos elementos distintos do grupo quociente $U(r)/\langle \overline{n}, \overline{p} \rangle$. Seja h um fator irredutível do polinômio $x^r - 1$ em $\mathbb{Z}_p[x]$. Pela proposição 2.6.4 o anel quociente $K = \mathbb{Z}_p[x]/(h)$ é um corpo finito. Denotaremos por G o subgrupo do grupo multiplicativo K^* gerado pelos elementos $\hat{x}^{m_i} + \hat{a}$, onde $1 \leq i \leq d$ e $a \in S$.

A maior parte da demonstração consiste em mostrar que o grupo G é "grande o suficiente". Mais precisamente:

$$|G| \geq n^{2\left\lfloor \sqrt{\frac{r-1}{d}} \right\rfloor}. \qquad (*)$$

Faremos isso construindo uma quantidade suficientemente grande de elementos de G. Com esse intuito introduzimos a aplicação que a cada função $e : S \longrightarrow \mathbb{N}$ associa a d-upla,

$$(\tilde{e}(\hat{x}^{m_1}), ..., \tilde{e}(\hat{x}^{m_d})),$$

onde $\tilde{e}$ é o polinômio definido por

$$\tilde{e}(y) = \Pi_{a \in S}(y + a)^{e(a)}$$

Sabemos que $\hat{x}$ denota a classe de x em $K = \mathbb{Z}_p[x]/(h)$. Portanto, as entradas desta d-upla são elementos de G e então

Vamos provar que esta aplicação associa d-uplas distintas a funções distintas. Sejam e_1 e e_2 funções de S em $\mathbb{N}$. Se as d-uplas associadas a e_1 e e_2 coincidirem, então

$$\tilde{e}_1(\hat{x}^{m_i}) = \tilde{e}_2(\hat{x}^{m_i}) \text{ em } K,$$

para todo $1 \leq i \leq d$. Porém, pela hipótese 4 e pelo lema 5.2.1,

$$\tilde{e}_1(\hat{x}^{m_i})^{n^k p^l} = \Pi_{a \in S}((\hat{x}^{m_i} + \hat{a})^{e_1(a)})^{n^k p^l} = \Pi_{a \in S}((\hat{x}^{m_i n^k p^l} + \hat{a})^{e_1(a)}) = \tilde{e}_1(\hat{x}^{m_i n^k p^l})$$

em $\mathbb{Z}_p[x]/(x^r - 1)$, para todo $k, l \geq 0$.

Mas como h é um fator de $x^r - 1$, qualquer igualdade válida para $\mathbb{Z}_p[x]/(x^r - 1)$ também é válida para $K = \mathbb{Z}_p[x]/(h)$, logo

$$\tilde{e_1}(\hat{x}^{m_i n^k p^l}) = \tilde{e_2}(\hat{x}^{m_i n^k p^l}) \text{ em } K.$$

Pondo $g(x) = \tilde{e_1}(x) - \tilde{e_2}(x)$, como $m_1, ..., m_d$ são representantes distintos de cada uma das classes de $U(r)/\langle \overline{n}, \overline{p} \rangle$, então dado um inteiro $0 < \alpha < r$, $\exists k, l \in \mathbb{N}$ tal que

$$\alpha \equiv m_i n^k p^l \mod r$$

Ou seja, $\alpha = m_i n^k p^l + tr$ para algum $t \in \mathbb{Z}$. Então

$$g(\hat{x}^\alpha) = g(\hat{x}^{m_i n^k p^l}(\hat{x}^r)^t) = g(\hat{x}^{m_i n^k p^l}) \text{ em } \mathbb{Z}_p[x]/(x^r - 1),$$

já que $\hat{x}^r = \hat{1}$ em $\mathbb{Z}_p[x]/(x^r - 1)$. Como h é um fator de $x^r - 1$, então

$$g(\hat{x}^\alpha) = g(\hat{x}^{m_i n^k p^l}) \text{ em } \mathbb{Z}_p[x]/(h) = K.$$

Como já sabemos $\tilde{e_1}(\hat{x}^{m_i n^k p^l}) = \tilde{e_2}(\hat{x}^{m_i n^k p^l})$ em K, então $\tilde{e_1}(\hat{x}^\alpha) = \tilde{e_2}(\hat{x}^\alpha)$ em K e consequentemente

$$g(\hat{x}^\alpha) = \tilde{e_1}(\hat{x}^\alpha) - \tilde{e_2}(\hat{x}^\alpha) = 0 \text{ em } K.$$

Considere o conjunto

$$R = \{\hat{x}^\alpha \mid 1 \leq \alpha \leq r - 1\} \subset K.$$

Como $h(x) \mid x^r - 1$, então $\hat{x}^r = 1$ em K. Contudo, r é primo, de modo que $\hat{x}$, pelo lema 2.5.1, tem ordem r em K^*. Em particular os elementos de R são todos distintos. Mas cada elementos de R é raiz de $g(y)$ em K. Pela proposição 2.6.3 podemos afirmar que $g(y)$ tem grau no mínimo $r - 1$ como polinômio de $\mathbb{Z}_p[y]$.

Supondo por hipótese que

$$\Sigma_{a \in S} e_1(a) \text{ e } \Sigma_{a \in S} e_2(a) \text{ são menores que } r - 1, \qquad (**)$$

então

$$g(y) = 0 \Rightarrow \tilde{e_1}(y) = \tilde{e_2}(y) \text{ em } \mathbb{Z}_p[y].$$

Entretanto, pela hipótese 2, sendo a e a' elementos distintos de S

$$(y+\overline{a})-(y+\overline{a'})=\overline{a}-\overline{a'}\neq 0 \text{ em } \mathbb{Z}_p[y].$$

Logo temos polinômios irredutíveis distintos $y+\overline{a}$ e $y+\overline{a'}$ em $\mathbb{Z}_p[y]$. Sendo assim, pelo teorema da fatoração única 2.6.2, $\tilde{e}_1(y)=\tilde{e}_2(y)$ só pode acontece, sob a hipótese $(**)$, quando as funções e_1 e e_2 são iguais. Em particular, o conjunto das d-uplas de elementos de G tem, pelo menos, tantos elementos quantas são as funções $e:S\to\mathbb{N}$ pra as quais

$$\Sigma_{a\in S}e(a)<r-1\Rightarrow e(1)+e(2)+\ldots+e(r)\leq r-2.$$

Contudo, esta equação linear tem

$$\binom{2r-2}{r}$$

soluções inteiras não negativas, de modo que

$$|G|^d\geq\binom{2r-2}{r}.$$

Combinando esta desigualdade com a hipótese 3 temos que

$$|G|^d\geq n^{2d\left\lfloor\sqrt{\frac{r-1}{d}}\right\rfloor}$$

concluindo a prova da equação $(*)$.

Considere os produtos

$$n^ip^j \text{ para } 0\leq i,j\leq\left\lfloor\sqrt{\frac{r-1}{d}}\right\rfloor,$$

em particular,

$$1\leq n^ip^j\leq n^{2\left\lfloor\sqrt{\frac{r-1}{d}}\right\rfloor},$$

$(***)$

já que $p\leq n$. Cada classe $\overline{n^ip^j}\in\mathbb{Z}_r$ pertence ao subgrupo $\langle\overline{n},\overline{p}\rangle$ de $U(r)$. Mas, como informado anteriormente, a ordem do subgrupo $\langle\overline{n},\overline{p}\rangle$ é $\frac{r-1}{d}$. Portanto, não pode haver mais do que $\frac{r-1}{d}$ elementos distintos da forma $\overline{n^ip^j}$ em $\mathbb{Z}_r$.

Por outro lado, como i e j variam a partir de 0, há um total de $\left(\left\lfloor\sqrt{\frac{r-1}{d}}\right\rfloor+1\right)^2$ pares

$$(i,j) \text{ com } 0\leq i,j\leq\left\lfloor\sqrt{\frac{r-1}{d}}\right\rfloor.$$

Isto significa que as classes $\overline{n^i p^j}$ *não podem ser todas distintas em* $\mathbb{Z}_r$*. Portanto, existem pares*

$$(k_1, l_1) \neq (k_2, l_2) \text{ tais que } n^{k_1} p^{l_1} \equiv n^{k_2} p^{l_2} \mod r$$

Digamos que $u = n^{k_1} p^{l_1}$ *e que* $w = n^{k_2} p^{l_2}$*. Suponhamos* $w \geq u$*. Como u é congruente a w módulo r, então podemos escrever* $w = u + qr$*, para algum inteiro positivo q. Mas,*

$$\hat{x}^{u+qr} = \hat{x}^u (\hat{x}^r - 1 + 1)^q = \hat{x}^u \text{ em } \mathbb{Z}_p[x]/(x^r - 1),$$

de modo que

$$\hat{x}^w = \hat{x}^u \text{ em } \mathbb{Z}_p[x]/(x^r - 1).$$

Como $h(x)$ *divide* $x^r - 1$ *e* $K = \mathbb{Z}_p[x]/(h)$*, concluímos que*

$$\hat{x}^{u+qr} = \hat{x}^u \text{ em } K.$$

$(****)$

Porém, pela parte 4 do teorema e pelo lema 5.2.1,

$$\Pi_{a \in S}(\hat{x}^{m_i} + \hat{a})^u = \Pi_{a \in S}(\hat{x}^{m_i} + \hat{a})^{n^{k_1} p^{l_1}} = \Pi_{a \in S}(\hat{x}^u)^{m_i} + \hat{a}.$$

de modo que a igualdade $(****)$ *implica que*

$$\Pi_{a \in S}(\hat{x}^{m_i} + \hat{a})^u = \Pi_{a \in S}(\hat{x}^{m_i} + \hat{a})^w.$$

Como o produto desses termos geram G, temos que $g^w = g^u$ *em K para todo* $g \in G$*. Em particular, todos os elementos de* $G \cup \{0\}$ *são soluções da equação polinomial* $y^w - y^u$ *em K.*

Contudo, a desigualdade $(***)$ *implica que*

$$\begin{cases} 1 \leq w \leq n^{2\left\lfloor \sqrt{\frac{r-1}{d}} \right\rfloor} \\ 1 - n^{2\left\lfloor \sqrt{\frac{r-1}{d}} \right\rfloor} \leq 1 - u \leq 0 \\ 0 \leq w - u \end{cases} \implies 1 \leq w - u + 1 \leq n^{2\left\lfloor \sqrt{\frac{r-1}{d}} \right\rfloor}$$

e combinando esta desigualdade com $(*)$*, obtemos*

$$|G| \geq (w - u) + 1,$$

ou seja, G tem no mínimo $w - u + 1$ *elementos.*

Como $G \subset K$, concluímos que o polinômio

$$y^w - y^u = y^u(y^{w-u} - 1)$$

tem, pelo menos, $w-u+1$ soluções não nulas em K, pois todos os elementos de G são soluções válidas. Como, pela proposição 2.6.3, sabemos que um polinômio terá no máximo o seu grau em raízes e se $w-u > 0$ teremos apenas $y = 1$ como solução não nula, logo

$$w - u = 0 \Rightarrow u = w \Rightarrow n^{k_1} p^{l_1} = n^{k_2} p^{l_2}.$$

Porém, $k_1 = k_2$ implica que $p^{l_1} = p^{l_2}$; donde $(k_1, l_1) = (k_2, l_2)$, o que contradiz a escolha destes pares. Logo $k_1 \neq k_2$. Só que, neste caso,

$$n^{k_1 - k_2} = p^{l_2 - l_1}$$

de modo que n tem que ser uma potência de p, como queríamos demonstrar.

□

Nesta primeira versão apresentada pelos autores, foi provado que o algoritmo tem um limite superior do custo de $O(\log_2^{12} n)$. Vários matemáticos revisaram o artigo em busca de torná-lo mais eficiente, o que foi bastante positivo, pois logo nos meses seguintes havia a proposta de diferentes versões para o teste e com limites mais baixos.

A última versão do teste AKS proposta por Lenstra e Pomerance (JR; POMERANCE, 2019) conseguiu alcançar uma complexidade de $O(\log_2^6 n)$. É o limite teórico mais baixo para o teste que não depende de conjecturas ou possua exceções.

Apesar dos progressos realizados sob o algoritmo AKS, seu tempo de execução ainda é significativamente mais lento do que os testes probabilísticos, portanto ainda não é utilizado nas principais soluções para criptografia.

6 PROGRAMAÇÃO

C++ é uma linguagem de programação orientada a objetos e de uso geral. Desde os anos 1990 é uma das linguagens comerciais mais populares, sendo bastante usada também no meio acadêmico por seu grande desempenho e base de utilizadores.

Então em C++ seque os testes de primalidade contidos no capítulo 5. Os sites (CARMODY, 2003) e (GPOULOSE, 2003) foram utilizados como base para elaboração dos algoritmos. A biblioteca NTL (SHOUP, 2016) foi utilizada no algoritmo AKS.

6.1 TESTES PROBABILÍSTICOS

6.1.1 Solovay e Strassen

Código-fonte 1 – Solovay e Strassen

```
#include <bits/stdc++.h>
#include <time.h>
using namespace std;

long long mulo(long long b, long long e, long long m)
{
    long long x = 1;
    long long y = b;
    while (e > 0)
    {
        if (e % 2 == 1)
            x = (x * y) % m;

        y = (y * y) % m;
        e = e / 2;
    }

    return x % m;
}

int Jacobi(long long a, long long n)
{
    if (!a)
```

```
24          return 0;
25
26      int ans = 1;
27      if (a < 0)
28      {
29          a = -a;
30          if (n % 4 == 3)
31              ans = -ans;
32      }
33
34      if (a == 1)
35          return ans;
36
37      while (a)
38      {
39          if (a < 0)
40          {
41              a = -a;
42              if (n % 4 == 3)
43                  ans = -ans;
44          }
45
46          while (a % 2 == 0)
47          {
48              a = a / 2;
49              if (n % 8 == 3 || n % 8 == 5)
50                  ans = -ans;
51
52          }
53
54          swap(a, n);
55
56          if (a % 4 == 3 && n % 4 == 3)
57              ans = -ans;
58          a = a % n;
59
60          if (a > n / 2)
61              a = a - n;
62
```

```
63      }
64
65      if (n == 1)
66          return ans;
67
68      return 0;
69  }
70
71  bool solovoyStrassen(long long p, int iter)
72  {
73      if (p < 2)
74          return false;
75      if (p != 2 && p % 2 == 0)
76          return false;
77      for (int i = 0; i < iter; i++)
78      {
79          long long a = rand() % (p - 1) + 1;
80          long long jacobian = (p + Jacobi(a, p)) % p;
81          long long m = mulo(a, (p - 1) / 2, p);
82
83          if (!jacobian || m != jacobian)
84              return false;
85      }
86      return true;
87  }
88
89  int main()
90  {
91    clock_t inicio, parar;
92      int iter;
93      long long num;
94      int f;
95      cout << "Solovay e Strassen" << endl;
96      cout << "Digite o numero de interacoes e em sequida o
            numero impar a ser testado:" << endl;
97      cin >> iter >> num;
98      for (;iter>=1;)
99    {
100     cout << "Digite a quatidade de numeros impares seguidos a
```

```
        serem testados: " << endl;
101     cin >> f;
102     for (int g=1; g<=f; g++)
103     {
104         inicio = clock();
105         if (solovoyStrassen(num, iter))
106             cout<< num << " primo"<<endl;
107         else
108             cout<< num << " composto"<<endl;
109         parar = clock();
110         cout << "Tempo de execucao " << parar-inicio << " ms "
                << endl;
111         num +=2;
112       }
113       cout << "Para encerrar digite 0 duas vezes, para
              continuar digite o numero de interacoes e depois o
              numero impar a ser testado: " << endl;
114     cin >> iter >> num;
115     }
116     return 0;
117 }
```

6.1.2 Miller-Rabin

Código-fonte 2 – Miller-Rabin

```
1  #include <iostream>
2  #include <cstring>
3  #include <cstdlib>
4  #include <time.h>
5  #define ll long long
6  using namespace std;
7
8  ll mulmod(ll a, ll b, ll mod)
9  {
10     ll x = 0,y = a % mod;
11     while (b > 0)
12     {
13         if (b % 2 == 1)
```

```
14          {
15              x = (x + y) % mod;
16          }
17          y = (y * 2) % mod;
18          b /= 2;
19      }
20      return x % mod;
21  }
22
23  ll modulo(ll base, ll exponent, ll mod)
24  {
25      ll x = 1;
26      ll y = base;
27      while (exponent > 0)
28      {
29          if (exponent % 2 == 1)
30              x = (x * y) % mod;
31          y = (y * y) % mod;
32          exponent = exponent / 2;
33      }
34      return x % mod;
35  }
36
37
38  bool Miller(ll p,int iteration)
39  {
40      if (p < 2)
41      {
42          return false;
43      }
44      if (p != 2 && p % 2==0)
45      {
46          return false;
47      }
48      ll s = p - 1;
49      while (s % 2 == 0)
50      {
51          s /= 2;
52      }
```

```
53      for (int i = 0; i < iteration; i++)
54      {
55          ll a = rand() % (p - 1) + 1, temp = s;
56          ll mod = modulo(a, temp, p);
57          while (temp != p - 1 && mod != 1 && mod != p - 1)
58          {
59              mod = mulmod(mod, mod, p);
60              temp *= 2;
61          }
62          if (mod != p - 1 && temp % 2 == 0)
63          {
64              return false;
65          }
66      }
67      return true;
68  }
69
70  int main()
71  {
72    clock_t inicio, parar;
73      int iter, f;
74      ll num;
75      cout << "Miller-Rabin" << endl;
76      cout << "digite o numero de interacoes e depois o numero
            impar a ser testado:" << endl;
77    cin >> iter >> num;
78    for (;iter>=1;)
79    {
80      cout << "Digite a quatidade de numeros impares seguidos a
            serem testados: " << endl;
81      cin >> f;
82      for (int g=1; g<=f; g++)
83      {
84        inicio = clock();
85          if (Miller(num, iter))
86              cout<< num << " primo"<<endl;
87          else
88              cout<< num << " composto"<<endl;
89          parar = clock();
```

```
90          cout << "Tempo de execucao " << parar-inicio << " ms "
                << endl;
91          num +=2;
92        }
93        cout << "Para encerrar digite 0 duas vezes, para
             continuar digite o numero de interacoes e depois o
             numero impar a ser testado: " << endl;
94      cin >> iter >> num;
95      }
96      return 0;
97  }
```

6.2 TESTES DETERMINÍSTICOS

6.2.1 Crivo de Eratóstenes

Código-fonte 3 – Crivo de Eratóstenes

```
1  #include <bits/stdc++.h>
2  #include <time.h>
3  using namespace std;
4
5  void Eratostenes(long long num)
6  {
7      bool prime[num+1];
8      memset(prime, true, sizeof(prime));
9      for (long long p=2; p*p<=num; p++)
10     {
11     if (prime[p] == true)
12       {
13       for (long long i=p*p; i<=num; i += p)
14       prime[i] = false;
15       }
16     }
17     for (long long p=2; p<=num; p++)
18     if (prime[p])
19     cout << p << " ";
20     cout << endl;
21 }
```

```
22 int main()
23 {
24   clock_t inicio, parar;
25   long long num;
26   int f;
27     cout << "Crivo de Eratostenes" << endl;
28   cout << "Digite o numero: " << endl;
29   cin >> num;
30
31   for (;num>=2;)
32   {
33     cout << "Digite a quatidade de numeros impares seguidos a
         serem testados: " << endl;
34     cin >> f;
35     for (int g=1; g<=f; g++)
36     {
37       cout << "Os numeros primos ate " << num << " sao: " <<
           endl;
38         inicio = clock();
39       Eratostenes(num);
40       parar = clock();
41       cout << "Tempo de execucao " << parar-inicio << " ms " <<
            endl;
42       num +=2;
43     }
44     cout << "Para encerrar digite 0, para continuar digite o
         numero: " << endl;
45     cin >> num;
46   }
47     return 0;
48 }
```

6.2.2 Agrawal, Kayal e Saxena

Código-fonte 4 – Agrawal, Kayal e Saxena

```
1 #include <stdio.h>
2 #include <time.h>
3 #include <math.h>
```

```
4  #include <NTL/ZZ.h>
5  #include <NTL/RR.h>
6  #include <NTL/ZZ_pE.h>
7  #include <NTL/ZZ_pEX.h>
8
9  using namespace std;
10 using namespace NTL;
11
12 int step0(ZZ n);
13 int step1(ZZ n);
14 long step2(ZZ n);
15 int step3(ZZ n, long r);
16 int step4(ZZ n, long r);
17 int step5(ZZ n, long r);
18 int step6();
19 ZZ  gcd(ZZ m, ZZ n);
20 ZZ  order(ZZ  r, ZZ a);
21 ZZ phi(ZZ n);
22
23 int main(void)
24 {
25   int k, f;
26   ZZ n;
27   long r;
28   clock_t inicio, parar;
29
30   cout << "AKS" << endl;
31   cout << "digite o numero impar a ser testado:" << endl;
32   cin >> n;
33
34   for (; n >= 1;)
35   {
36     cout << "Digite a quatidade de numeros impares seguidos a
         serem testados: " << endl;
37     cin >> f;
38     for (int g = 1; g <= f; g++)
39     {
40       inicio = clock();
41       k = step0(n);
```

```
42        if (k)
43        {
44          if (k == 1)
45          {
46            cout << n << " composto\n";
47          }
48          else
49          {
50            cout << n << " primo\n";
51          }
52        }
53        else
54        {
55          k = step1(n);
56          if (k)
57          {
58            cout << n << " composto\n";
59          }
60          else
61          {
62            r = step2(n);
63            k = step3(n, r);
64            if (k)
65            {
66              cout << n << " composto\n";
67            }
68            else
69            {
70              k = step4(n, r);
71              if (k)
72              {
73                cout << n << " primo\n";
74              }
75              else
76              {
77                k = step5(n, r);
78                if (k)
79                {
80                  cout << n << " composto\n";
```

```
81                }
82                else
83                {
84                  k = step6();
85                  if (k)
86                  {
87                    cout << n << " primo\n";
88                  }
89                }
90              }
91            }
92          }
93        }
94        parar = clock();
95        cout << "Tempo de execucao " << parar - inicio << " ms "
              << endl;
96        n += 2;
97      }
98      cout << "Para encerrar digite 0, para continuar digite o
            numero impar a ser testado: " << endl;
99      cin >> n;
100   }
101 }
102
103
104 ZZ gcd(ZZ m, long n)
105 {
106   ZZ k, z;
107   z = 0;
108   z = n + z;
109   if (z < m)
110   {
111     swap(z, m);
112   }
113
114   while (m % z != 0)
115   {
116     k = m % z;
117     m = z;
```

```
118       z = k;
119     }
120     return z;
121 }
122 
123 ZZ order(ZZ a, long r)
124 {
125     ZZ k, o, z;
126     o = 1;
127     z = 1;
128     k = a;
129     z = r + z;
130 
131     while (k != 1)
132     {
133       k *= a; k %= z; o++;
134     }
135     return o;
136 }
137 
138 ZZ phi(long x)
139 {
140     ZZ n, ph, p; ph = 1;
141     n = 0; n = n + x;
142     int k;
143 
144     for (PrimeSeq s; n > 1; )
145     {
146       p = s.next();
147       k = 0;
148       while (n % p == 0)
149       {
150         k++; n /= p;
151       }
152       if (k > 0)
153         ph *= power(p, k - 1) * (p - 1);
154     }
155     return ph;
156 }
```

```
157
158 int step0(ZZ n)
159 {
160
161   long p;
162
163   PrimeSeq s;
164
165   p = s.next();
166   while (p && p < 2000)
167   {
168     if ((n % p) == 0)
169     {
170       return (1 + (n == p));
171     }
172     p = s.next();
173   }
174   return 0;
175 }
176
177 int step1(ZZ n)
178 {
179   RR A, B, N, log_2000_n;
180   long b = 2;
181   log_2000_n = log(n) / log(2000);
182   N = to_RR(n);
183
184   while (b < log_2000_n)
185   {
186     B = 1.0 / b;
187     A = pow(N, B);
188     if ((A - to_RR(FloorToZZ(A))) == 0.0)
189     {
190       if (power(FloorToZZ(A), b) == n)
191         return 1;
192     }
193     b++;
194   }
195   return 0;
```

```
196 }
197
198 long step2(ZZ n)
199 {
200   RR log2n, l;
201   ZZ k;
202   long r;
203   log2n = NumBits(n);
204   l = power(log2n, 2);
205   k = FloorToZZ(l);
206
207   for (r = 2;; r++)
208   {
209     if (gcd(n, r) == 1)
210     {
211       if (order(n, r) > k)
212       {
213         return r;
214       }
215     }
216   }
217 }
218
219 int step3(ZZ n, long r)
220 {
221   long a;
222   for (a = 2002; ++a < r;)
223   {
224     if ((gcd(n, a) % n) > 1)
225       return 1;
226   }
227   return 0;
228 }
229
230 int step4(ZZ n, long r)
231 {
232   if (n <= r)
233     return 1;
234   return 0;
```

```
235 }
236
237 int step5(ZZ n, long r)
238 {
239   ZZ l;
240   long a;
241   NTL::ZZ_p::init(n);
242   ZZ_pX polymod(r, 1);
243   polymod -= 1;
244   ZZ_pXModulus mod(polymod);
245   ZZ_pX RHS(1, 1);
246   PowerMod(RHS, RHS, n, mod);
247   l = FloorToZZ(sqrt(to_RR(phi(r))) * NumBits(n));
248
249   for (a = 1; a <= l; a++)
250   {
251     ZZ_pX LHS(1, 1);
252     LHS += a;
253     PowerMod(LHS, LHS, n, mod);
254     LHS -= a;
255     if (LHS != RHS)
256       return 1;
257   }
258   return 0;
259 }
260
261 int step6()
262 {
263   return 1;
264 }
```

7 RESULTADOS

Para facilitar o entendimento e visualização da diferença entre os testes de primalidade aqui discutidos, comparamos as saídas dos programas com 240 números ímpares, conferindo o resultado com a lista de primos do site (ASSOCIéS, 20–?) e observando o tempo de execução. Todos os testes foram efetuados no mesmo computador e nas mesmas condições para mitigar os fatores externos ao algoritmo. Os testes de Solovay Strassen e Miller-Rabin foram efetuados sempre com 5 interações. O tempo será relatado em milissegundos, quando o teste informar que o número é primo teremos um *p* antecedendo o tempo do teste.

Tabela 1 – Comparação de Testes de Primalidade

(continua)

Número	Solovay Strassen	Miller-Rabin	Eratóstenes	AKS
8911	0	15	2515	4
8913	0	0	2250	5
8915	16	0	2203	4
8917	0	0	2219	4
8919	0	0	2313	5
8921	0	15	2422	4
8923	P 0	P 0	P 2437	P 6008
8925	0	0	2312	4
8927	0	0	2375	4
8929	P 0	P 0	P 2156	P 5877
8931	0	0	2265	3
8933	P 16	P 0	P 2328	P 8497
8935	0	0	2375	4
8937	0	0	2250	4
8939	15	0	2546	7
8941	P 0	P 0	P 2500	P 3506
8943	0	0	2406	4
8945	15	0	1953	5
8947	16	15	1641	6
8949	0	0	2094	7

Tabela 1 – Comparação de Testes de Primalidade

(continuação)

Número	Solovay Strassen	Miller-Rabin	Eratóstenes	AKS
8951	P 0	P 0	P 2578	P 3559
8953	15	0	2422	7
8955	0	0	2156	3
8957	0	16	2171	5
8959	16	0	2343	8
8961	0	0	2171	6
8963	P 16	P 15	P 2281	P 4416
8965	0	0	2265	4
8967	0	0	2484	5
8969	P 16	P 0	P 1984	P 4122
8971	P 0	P 0	P 2115	P 3302
8973	0	15	1640	7
8975	15	0	1719	4
8977	0	0	1734	7
8979	0	15	1875	4
8981	0	0	1922	4
8983	15	0	1812	4
8985	0	0	1766	9
8987	0	0	1735	4
8989	0	15	1734	4
8991	16	0	1749	12
8993	0	16	1781	5
8995	0	0	1750	4
8997	0	0	1782	4
8999	P 0	P 0	P 1719	P 3412
9001	P 0	P 0	P 1578	P 4531
9003	0	16	1469	2
9005	0	0	1421	2
9007	P 0	P 0	P 1406	P 2563

Tabela 1 – Comparação de Testes de Primalidade

(continuação)

Número	Solovay Strassen	Miller-Rabin	Eratóstenes	AKS
9009	0	0	1422	2
9011	P 15	P 0	P 1438	P 2936
9013	P 16	P 16	P 1407	P 2659
9015	0	0	1422	3
9017	0	0	1391	3
9019	15	0	1438	6
9021	0	0	1407	2
9023	0	16	1422	5
9025	16	0	1562	3
9027	15	0	1814	6
9029	P 0	P 16	P 2301	P 2038
9031	0	0	3702	4
9033	0	0	1012	3
9035	16	0	951	5
9037	0	0	719	3
9039	0	16	734	5
9041	P 15	P 0	P 734	P 2837
9043	P 0	P 16	P 750	P 2515
9045	0	16	736	3
9047	0	0	795	4
9049	P 15	P 0	P 847	P 3145
9051	16	0	848	3
9053	0	0	732	4
9055	0	0	735	6
9057	0	16	744	4
9059	P 0	P 0	P 788	P 3998
9061	16	0	771	3
9063	0	15	1095	2
9065	0	16	767	3

Tabela 1 – Comparação de Testes de Primalidade

(continuação)

Número	Solovay Strassen	Miller-Rabin	Eratóstenes	AKS
9067	P 16	P 0	P 790	P 2844
9069	0	0	761	3
9071	0	15	764	5
9073	15	0	780	5
9075	16	0	772	5
9077	15	15	767	6
9079	0	0	773	5
9081	0	0	779	3
9083	16	0	786	1
9085	0	15	783	4
9087	0	0	777	5
9089	16	0	787	3
9091	P 0	P 16	P 791	P 3219
9093	0	0	784	3
9095	0	0	791	5
9097	16	0	795	4
9099	0	0	786	4
9101	0	15	783	5
9103	P 16	P 0	P 777	P 2970
9105	0	16	788	3
9107	0	0	781	9
9109	P 15	P 0	P 777	P 3333
444556667	P 0	P 0	ERRO	P 473679
444556669	P 15	P 0	ERRO	P 397386
444556671	0	0	ERRO	3
444556673	0	16	ERRO	2
444556675	15	0	ERRO	5
444556677	0	0	ERRO	3
444556679	0	16	ERRO	3

Tabela 1 – Comparação de Testes de Primalidade

(continuação)

Número	Solovay Strassen	Miller-Rabin	Eratóstenes	AKS
444556681	16	0	ERRO	3
444556683	0	0	ERRO	3
444556685	0	16	ERRO	5
444556687	P 16	P 0	ERRO	P 443416
444556689	0	0	ERRO	5
444556691	0	16	ERRO	5
444556693	16	16	ERRO	8
444556695	15	15	ERRO	13
444556697	47	0	ERRO	12
444556699	0	16	ERRO	13
444556701	16	16	ERRO	8
444556703	0	0	ERRO	13
444556705	16	0	ERRO	12
444556707	0	15	ERRO	9
444556709	15	0	ERRO	9
444556711	16	16	ERRO	1889
444556713	0	0	ERRO	7
444556715	16	16	ERRO	7
444556717	16	16	ERRO	6
444556719	0	16	ERRO	7
444556721	15	0	ERRO	7
444556723	16	0	ERRO	6
444556725	0	15	ERRO	10
444556727	16	0	ERRO	11
444556729	P 16	P 15	ERRO	P 345184
444556731	0	16	ERRO	8
444556733	P 15	P 0	ERRO	P 489601
444556735	16	16	ERRO	6
444556737	0	0	ERRO	10

Tabela 1 – Comparação de Testes de Primalidade

(continuação)

Número	Solovay Strassen	Miller-Rabin	Eratóstenes	AKS
444556739	16	16	ERRO	8
444556741	P 15	P 0	ERRO	P 441508
444556743	16	0	ERRO	10
444556745	0	16	ERRO	6
444556747	16	16	ERRO	7
444556749	0	0	ERRO	5
444556751	0	16	ERRO	5
444556753	15	0	ERRO	5
444556755	16	16	ERRO	8
444556757	P 0	P 15	ERRO	P 407084
444556759	16	15	ERRO	5
444556761	16	0	ERRO	7
444556763	0	0	ERRO	5
444556765	15	16	ERRO	8
444556767	16	16	ERRO	6
444556769	0	0	ERRO	5
444556771	16	15	ERRO	5
444556773	0	0	ERRO	5
444556775	0	15	ERRO	8
444556777	15	16	ERRO	9
444556779	16	15	ERRO	8
444556781	0	16	ERRO	1144
444556783	16	0	ERRO	4
444556785	0	16	ERRO	3
444556787	0	15	ERRO	4
444556789	15	0	ERRO	6
444556791	0	16	ERRO	4
444556793	P 0	P 0	ERRO	P 452300
444556795	16	16	ERRO	5

Tabela 1 – Comparação de Testes de Primalidade

(continuação)

Número	Solovay Strassen	Miller-Rabin	Eratóstenes	AKS
444556797	0	16	ERRO	8
444556799	16	16	ERRO	8
444556801	15	31	ERRO	5
444556803	0	15	ERRO	5
444556805	15	0	ERRO	7
444556807	0	15	ERRO	1174
444556809	0	16	ERRO	5
444556811	16	0	ERRO	5
444556813	15	0	ERRO	5
444556815	0	0	ERRO	7
444556817	15	16	ERRO	4
444556819	16	16	ERRO	5
444556821	0	15	ERRO	7
444556823	16	16	ERRO	1573
444556825	15	0	ERRO	12
444556827	0	0	ERRO	11
444556829	15	0	ERRO	1713
444556831	0	16	ERRO	6
444556833	15	15	ERRO	6
444556835	16	0	ERRO	8
444556837	0	0	ERRO	10
444556839	0	0	ERRO	9
444556841	P 15	P 0	ERRO	P 1839381
444556843	P 16	P 15	ERRO	P 318435
444556845	0	16	ERRO	8
444556847	16	16	ERRO	12
444556849	0	15	ERRO	13
444556851	16	0	ERRO	11
444556853	15	0	ERRO	12

Tabela 1 – Comparação de Testes de Primalidade

(continuação)

Número	Solovay Strassen	Miller-Rabin	Eratóstenes	AKS
444556855	0	0	ERRO	8
444556857	15	15	ERRO	12
444556859	16	16	ERRO	13
444556861	16	15	ERRO	9
444556863	16	16	ERRO	11
444556865	0	0	ERRO	12
3012345817	16	0	ERRO	12
3012345819	15	15	ERRO	11
3012345821	0	15	ERRO	12
3012345823	0	16	ERRO	12
3012345825	0	15	ERRO	8
3012345827	P 0	P 16	ERRO	P 2031549
3012345829	16	15	ERRO	4678
3012345831	16	16	ERRO	18
3012345833	15	0	ERRO	17
3012345835	16	16	ERRO	15
3012345837	0	0	ERRO	17
3012345839	P 0	P 0	ERRO	P 1591060
3012345841	P 0	P 15	ERRO	P 1677596
3012345843	0	16	ERRO	13
3012345845	16	16	ERRO	11
3012345847	15	16	ERRO	12
3012345849	16	16	ERRO	12
3012345851	P 16	P 0	ERRO	P 1685518
3012345853	15	0	ERRO	14
3012345855	0	15	ERRO	21
3012345857	0	0	ERRO	17
3012345859	0	15	ERRO	13
3012345861	16	0	ERRO	17

Tabela 1 – Comparação de Testes de Primalidade

(conclusão)

Número	Solovay Strassen	Miller-Rabin	Eratóstenes	AKS
3012345863	16	15	ERRO	15
3012345865	15	0	ERRO	16
3012345867	0	0	ERRO	17
3012345869	0	16	ERRO	16
3012345871	P 0	P 15	ERRO	P 1710420
3012345873	0	0	ERRO	16
3012345875	0	15	ERRO	18
3012345877	P 0	P 16	ERRO	P 1760066
3012345879	16	0	ERRO	14
3012345881	16	16	ERRO	15
3012345883	P 15	P 15	ERRO	P 1642452
3012345885	16	0	ERRO	18
3012345887	16	15	ERRO	17
3012345889	15	0	ERRO	17
3012345891	0	16	ERRO	17
3012345893	0	0	ERRO	17
3012345895	0	0	ERRO	17

Fonte: Elaborado pelo autor

A mosca bate as asas a cada 3 ms, uma abelha em 5 ms, a Lua viaja ao redor da Terra 2 ms mais lentamente a cada ano conforme sua órbita gradualmente se alarga. Na ciência computacional, um intervalo entre 1 e 10 ms é conhecido como jiffy e é definida como o tempo de um clique, sendo considerado com um erro aceitável.

Como os algoritmos probabilísticos foram executados com 5 interações, a probabilidade de um número composto ser dado como primo é $\frac{\ln n}{2^5}$ para Solovay e Strassen e $\frac{1}{4^5}$ para Miller-Rabin, portanto, nos testes efetuados não encontramos nenhum falso primo. O tempo gasto por esses testes foram bem baixos por conta da quantidade de interações efetuadas.

O crivo de Eratóstenes não está copilando para números grandes, provavelmente por limitação de bits, dado pela classe de algumas etapas do algoritmo.

Já o AKS durante o teste se mostrou muito eficiente para números compostos e, por conta das várias etapas necessárias para confirmar a primalidade para números primos, ainda está bem mais lento.

Essas implementações nos algoritmos podem ser otimizadas com intuito de dar melhor simplicidade e agilidade aos testes, mas a matemática base de cada algoritmo não muda. Em pesquisas verifiquei que outras implementações mais enxutas do AKS já existem, mas isso poderá fazer parte de um outro trabalho.

8 CONCLUSÕES E TRABALHOS FUTUROS

Vimos no decorrer desse trabalho um pouco da história dos números primos, sua aplicação na computação e como testamos se um número é primo para poder utilizá-lo. Trabalhamos com 4 testes de primalidade e comparamos seus resultados.

Pretendo que esse trabalho seja lido pelos professores e adaptado conforme sua realidade em sala de aula com intuito de mostrar que muitas vezes utilizamos a matemática sem saber, criando interesse e empatia do aluno com a matéria.

Como trabalho futuro, propõe-se o estudo de outros teste de primalidade e o aprimoramento dos códigos aqui apresentados.

Termino esse trabalho com uma frase que meu orientador gosta muito de dizer e que marcou os dias que acordei 3h da manhã para estudar.

"Matemático é uma máquina que transforma café em teorema."

Alfréd Rényi

REFERÊNCIAS

AKEL, A. **A IMPORTÂNCIA dos números primos**. 2016. Disponível em: <https://medium.com/unidades-imaginárias/a-importância-dos-números-primo-1249a54cc57e>. Acesso em: 28 mar. 2019.

ALFORD, W. R.; GRANVILLE, A.; POMERANCE, C. There are infinitely many carmichael numbers. **Annals of Mathematics**, JSTOR, v. 139, n. 3, p. 703–722, 1994.

ASSOCIéS, L. composants. **Prime I.T.** 20–? Disponível em: <http://compoasso.free.fr/primelistweb/page/prime/liste_online.php>.

CARMODY, P. **The AKS "PRIMES in P"Algorithm Resource**. 2003. Disponível em: <http://fatphil.org/maths/AKS/#Analyses>. Acesso em: 03 maio 2019.

COUTINHO, S. C. **Primalidade em tempo polinomial**. [S.l.]: Sociedade Brasileira de Matemática, 2004.

DIETZFELBINGER, M. **Primality testing in polynomial time**: from randomized algorithms to"primes is in p". [S.l.]: Springer, 2004. v. 3000.

GALLIER, J.; QUAINTANCE, J. **Notes on Primality Testing And Public Key Cryptography Part 1**: Randomized algorithms miller–rabin and solovay–strassen tests. [S.l.], 2019. v. 19104.

GPOULOSE. **AKS.cpp**. 2003. Disponível em: <https://www.gpoulose.com/gc/AKS_cpp.txt>. Acesso em: 02 abr. 2019.

HEFEZ, A. **Aritmética**. [S.l.]: Sociedade Brasileira de Matemática, 2016.

JR, H. W. L.; POMERANCE, C. B. Primality testing with gaussian periods. **Journal of the European Mathematical Society**, 2019.

KATZ, J.; MENEZES, A. J.; OORSCHOT, P. C. V.; VANSTONE, S. A. **Handbook of applied cryptography**. [S.l.]: CRC press, 1996.

MAIER, R. R. **Teoria dos números**. [S.l.]: Universidade de Brasília-Departamento de Matematica-IE, 2005.

MARTINEZ, C. G. M. F. **Teoria dos Números**: um passeio com primos e outros números familiares pelo mundo inteiro. [S.l.]: Instituto de Matemática Pura e Aplicada, 2015.

MERSENNE. **GIMPS**. 1996. Disponível em: <https://www.mersenne.org/>. Acesso em: 28 mar. 2019.

NASCIMENTO, M. C. d.; FEITOSA, H. d. A. **Elementos da teoria dos números**. São Paulo: Cultura, 2009.

NOé, M. **Numeros Primos**. 2017. Disponível em: <https://alunosonline.uol.com.br/matematica/numeros-primos-.html>. Acesso em: 28 mar. 2019.

RIESEL, H. **Prime numbers and computer methods for factorization**. [S.l.]: Springer Science & Business Media, 2012. v. 126.

ROUSSEAU, C.; SAINT-AUBIN, Y. **Matemática e Atualidade**. Rio de Janeiro: Sociedade Brasileira de Matemática, 2015. v. 1.

ROUSSEAU, C.; SAINT-AUBIN, Y.; ANTAYA, H.; ASCAH-COALLIER, I.; HAMILTON, C. **Mathematics and technology**. [S.l.]: Springer, 2008.

SANTOS, J. P. de O. **Introdução à Teoria dos Números**. [S.l.]: Instituto de Matemática Pura e Aplicada, 1998.

SHOUP. **NTL**: A library for doing number theory. 2016. Disponível em: <https://www.shoup.net/ntl/>. Acesso em: 25 maio 2019.