INVESTIGANDO FATORES PRIMOS COM TRINCAS PITAGÓRICAS

INVESTIGATING PRIME FACTORS WITH PYTHAGOREAN TRIPLES

Alessandro Firmiano de Jesus[*]
João Paulo Martins dos Santos[**]

RESUMO

De conceitos acessíveis à maioria dos professores de matemática do ensino médio, este artigo propõe uma estratégia de fatoração numérica baseada na primalidade de $p > 2$ indicada pela unicidade de trincas pitagóricas (p, b, c). As demonstrações são diretas e de abordagens diferenciadas para citar, na área da Teoria dos Números, seus ilustres personagens centrais e tópicos consagrados, tais como: Sophie Germain, Fermat, Mersenne, Números de Euclides, Conjecturas, Primos Gêmeos e Primos Pitagóricos. Nessa estratégia de decomposição proposta, técnicas triviais são complementadas com algumas inovações. A linguagem Python foi essencial para reunir programação com estratégia de fatoração e, para evidenciar que o algoritmo produz resultados corretos, foram submetidos testes de primalidade em uma sequência de grandes números primos da literatura que seguem dispostos em progressão aritmética.

Palavras-chaves: Ensino de Matemática. Fatoração. Código Python. Teste de Primalidade.

ABSTRACT

Grounding on concepts accessible to mathematics high school teachers, this work proposes a factoring strategy based on the primality of $p > 2$ as indicated by the unicity of the pythagorean triple (p, b, c). The proofs are straightforward and supported by different approaches linked with celebrated personalities and topics in the field of Number Theory, such as: Sophie Germain, Fermat, Mersenne, Euclidean numbers, Conjectures, Twin primes and Pythagorean primes. In this proposed decomposition strategy, standard techniques are complemented with some new ideas. Python language was instrumental to consolidate computer programming with factoring strategies and, in order to demonstrate that the algorithm yields correct results, primality tests were applied on a sequence of large primes in arithmetic progression identified in the literature.

Keywords: Math Teaching. Factorization. Python Coding. Primality Test.

Introdução

A versatilidade do Teorema de Pitágoras $a^2 + b^2 = c^2$ avança a mera resolução de problemas geométricos. Neste artigo, notáveis características das trincas pitagóricas (a, b, c) são

[*] Doutor em Ciências pelo SHS-USP e Prof. Assoc. IV de Matemática na Academia da Força Aérea – AFA, Pirassununga-SP. firmianoafj@fab.mil.br
[**] Doutor em Ciências pelo SHS-USP e Prof. Adj. II de Matemática na Academia da Força Aérea – AFA, Pirassununga-SP. jp2@usp.br

empregadas na verificação da primalidade de inteiros positivos ou de sua fatoração numérica. Para isto, inicialmente observe que todo número natural $a > 2$ poderá ser cateto em uma ou mais trincas pitagóricas (a, b, c) (FIRMIANO, et al. 2020). Ou seja, em função da paridade do cateto a, considere as trincas:

$$\begin{cases} a = 2n+1 \\ b = 2n^2 + 2n \\ c = b+1 \end{cases} \quad \text{ou} \quad \begin{cases} a = 2n \\ b = n^2 - 1 \\ c = b+2 \end{cases} \tag{1}$$

É possível também generalizar as trincas pitagóricas para o caso em que a hipotenusa coincide com $c = b + k$, sendo $1 \leq k < a$ inteiro. Para isto, relacione os catetos a e b partindo da expressão: $a^2 = (b+k)^2 - b^2 = \underbrace{2bk}_{par} + k^2$, ou seja, faça

$$b = \frac{a^2 - k^2}{2k} \in \mathbb{N} \tag{2}$$

Uma vez que o cateto a e o parâmetro $k < a$ possuem a mesma paridade, no caso a sendo um número par $a = 2n$, então $k = 2m$ para algum $n, m \in \mathbb{N}$ e $1 \leq m < n$. Substituindo em (2),

$$b = \frac{4n^2 - 4m^2}{4m} = \frac{n^2 - m^2}{m} \Rightarrow m|(n-m)(n+m) \tag{3}$$

Para o caso em que a é ímpar, ou seja, $a = 2n + 1$, então $k = 2m + 1$ para algum $n, m \in \mathbb{N}$ e $0 \leq m < n$. Ainda em (2),

$$b = \frac{(2n+1)^2 - (2m+1)^2}{2(2m+1)} = \frac{2(n-m)(n+m+1)}{2m+1} \Rightarrow (2m+1)|(n-m)(n+m+1) \tag{4}$$

Dessa forma, as divisibilidades (3) e (4) definem critérios de formação das seguintes trincas pitagóricas:

$\left(a = 2n, b = \frac{n^2-m^2}{m}, c = b + 2m\right)$ para a par ou

$\left(a = 2n+1, b = \frac{2(n-m)(n+m+1)}{2m+1}, c = b + 2m + 1\right)$ para a ímpar.

O critério (3) sempre é válido para $m = 1$, ou seja, para $k = 2$. Logo, todo número par $2n$ é associado à trinca $(2n, n^2 - 1, n^2 + 1)$. O critério (4) sempre é válido para $m = 0$, ou seja, para $k = 1$. Logo, todo ímpar $2n + 1$ é associado a uma trinca $(2n + 1, 2n^2 + 2n, n^2 + 2n + 1)$ conforme enunciados na equação (1).

Ao investigar outros valores que satisfazem o critério (4), o primeiro ímpar $k_0 > 1$ possivelmente encontrado é, naturalmente, um número primo. Pois, se k_0 fosse composto, todos os seus fatores, que também dividem a^2 e são menores do que k_0, seriam identificados antes,

logo, o número k_0 só pode ser um fator primo que divide a^2. Esta simples propriedade reforça o contexto da terminologia "primo", pois, no ordenamento crescente dos números ímpares, o número $k_0 > 1$ é o "primeiro" a ser identificado como um fator de a. Logo, é por esta simples razão que o fator k_0 é chamado de número primo.

Apresentando um exemplo numérico, considere $n = 12$, então os valores de $k = 2m$ definidos por todos os valores m que satisfazem o critério (3) são 2, 4, 6, 8, 12, 16 e 18. Assim, com o cateto $a = 24$ são formados apenas sete trincas pitagóricas: $(24{,}143{,}145)$, $(24{,}70{,}74)$, $(24{,}45{,}51)$, $(24{,}32{,}40)$, $(24{,}18{,}30)$, $(24{,}10{,}26)$ e $(24{,}7{,}25)$. No entanto, os valores $k = 2m + 1$ definidos pelos m que satisfazem o critério (4) são apenas $k = 1$ e o primo $k = 5$ para definirem, com o cateto $a = 25$, as respectivas trincas pitagóricas $(25{,}312{,}313)$ e $(25{,}60{,}65)$[1].

Conforme demonstrado em (FIRMIANO, *et al*. 2020) se $p > 2$ é um número primo, então existirá uma única trinca pitagórica na forma $(p, b_1, b_1 + k)$, logo $k = 1$, e uma única trinca na forma $(2p, b_2, b_2 + k)$, sendo $k = 2$. E ainda, a recíproca é verdadeira, ou seja, se existe apenas um único valor ímpar de k em $(a, b, b + k)$ satisfazendo o critério (4), então $k = 1$ e o cateto a é um número primo. Analogamente, se existe um único $k = 2m$ que satisfaz (3), então $k = 2$ e $a = 2p$, para p um número primo.

A prova destes resultados segue baseada no fato de que o parâmetro k sempre divide a^2, conforme sugere a equação (2). Assim, é possível afirmar, por exemplo, que $(17{,}144{,}145)$ e $(34{,}288{,}290)$ são as únicas trincas pitagóricas que possuem os respectivos catetos $a = 17$ ou $a = 2 \cdot 17 = 34$.

Outro resultado notável é a existência de certos valores de números primos q que definem, simultaneamente, uma única trinca pitagórica na forma $(2q + 1, b_1, b_1 + 1)$ e uma única na forma $(2q, b_2, b_2 + 2)$. Por exemplo, se $q = 11$, então $(23{,}264{,}265)$ é a única trinca pitagórica com $a = 2 \cdot 11 + 1 = 23$ e $(22{,}120{,}122)$ é a única trinca com o cateto $a = 2 \cdot 11 = 22$. No caso geral, esta dupla de trincas especiais será obtida sempre que $2q + 1$ e q, são ambos números primos. Ou seja, sempre que q for um **Número Primo de Sophie Germain**[2].

A ESTRATÉGIA DA DECOMPOSIÇÃO

Neste trabalho, a decomposição de um dado número $r_1 = 2n_1 + 1$, através da identificação do menor primo $p_1 > 2$ que divide r_1, baseia-se no critério de divisibilidade (4). Assim, quando se determina o menor valor $m_1 > 1$ tal que $(n_1 - m_1)(n_1 + m_1 + 1) \equiv 0 \bmod (2m_1 + 1)$, tem-se

[1] Pode ser provado que se $a = p^n$, para $p > 2$ primo e n natural, então a será cateto menor em exatamente n trincas pitagóricas. Caso $a = 2^n$, então a é cateto em exatamente $(n - 1)$ trincas pitagóricas.

[2] Estes primos são famosos porque Sophie Germain (1776-1831) provou que o Último Teorema de Fermat é verdadeiro para estes números. A questão da existência de infinitos destes números primos ainda é uma conjectura.

que $p_1 = 2m_1 + 1$ define o menor fator primo de r_1. Se $p_1 < r_1$, então o processo, de forma recursiva, determina $m_2 \geq m_1$ para definir $p_2 = 2m_2 + 1$ como sendo o menor fator primo de $r_2 = \frac{r_1}{p_1}$. Assim, após as iterações necessárias, todos os fatores primos de r_1 são obtidos.

Caso o algoritmo identifique que o critério (4) não é satisfeito para $0 < m_i < n_i$, então o número ímpar $r_i = 2n_i + 1$ é identificado como sendo um número primo, pois existirá uma única trinca pitagórica $(a, b, b + k)$ tal que $a = r_i$ e $k = 1$.

É fato que o proposto algoritmo de decomposição de n considera operações aritméticas envolvendo n^2. Assim, algumas estratégias computacionais foram implementadas para acelerar a investigação do menor fator primo de um inteiro positivo n_0 qualquer.

Estratégia 1 – No primeiro momento, o algoritmo identifica as maiores potências de 2 ou de 3 que dividem o número composto n_0, restando, assim, apenas possíveis fatores primos $p > 3$ a serem investigados.

Estratégia 2 – No esquema recursivo, apenas valores ímpares menores que $\sqrt{n_0}$ são testados no critério (4). Após a identificação do primeiro fator primo $5 \leq k_1 \leq \sqrt{n_0}$, o teste passa a ser realizado em $n_1 = \frac{n_0}{k_1}$, no entanto, com a verificação do critério (4) para $k_1 \leq k_2 \leq \sqrt{n_1}$. E assim, sucessivamente para $n_i = \frac{n_{i-1}}{k_i}$, investigam-se os fatores $k_i \leq k_{i+1} \leq \sqrt{n_i}$ até obter a divisão $\frac{n_i}{k_{i+1}} = 1$.

Estratégia 3 – Aplicar o fato de que os números primos $p > 3$ são da forma $p = 6m \pm 1$, com $m \in \mathbb{N}$.

Esta afirmação é uma consequência direta do seguinte lema:

Lema 1 Considere A, B e C uma sequência qualquer de ímpares consecutivos, então, exatamente um desses naturais é divisível por 3.

Demonstração

Sejam $A = 2m - 3;\ B = 2m - 1$ e $C = 2m + 1$, para algum natural $m > 1$. Assim,

$$A \cdot B \cdot C = (2m - 3)(2m - 1)(2m + 1) = 8m^3 - 12m^2 - 2m + 3$$

$$= \underbrace{6m^3 - 12m^2 + 3}_{parcela\ 1} + \underbrace{2m^3 - 2m}_{parcela\ 2}$$

A parcela 2 dada por $2m^3 - 2m = 2m(m^2 - 1) = 2\ \underbrace{(m-1)m(m+1)}_{3\ fatores\ consecutivos}$, e a parcela 1 são ambas divisíveis por 3 para todo $m > 1$, assim, segue que $3|(A \cdot B \cdot C)$. Desta forma, A, B ou C é o número ímpar divisível por 3. Considere agora que $3|A$, então $\begin{cases} B = A + 2 \equiv 2 \bmod 3 \\ C = A + 4 \equiv 1 \bmod 3 \end{cases}$, ou seja, $3 \nmid B$ e $3 \nmid C$. □□□

Com este Lema 1, é possível verificar que todo primo $p > 3$ é da forma $p = 6m \pm 1$. Para isto, numa sequência: $A = 2s - 3$; $B = 2s - 1$ e $C = 2s + 1$, considere $s > 2$ tal que $3|A$, isto é, $3|(2s - 3)$, logo, o ímpar A não é primo e $3|2s$. E ainda, $3 \nmid B$, $3 \nmid C$, mas $3|(C + 2)$.

Uma vez que, $6|2s$, e $2s = B + 1 = C - 1$, segue que, $\begin{cases} 6|(B+1) \\ 6|(C-1) \end{cases}$, ou seja, $\begin{cases} B = 6m - 1 \\ C = 6m + 1 \end{cases}$.

Nesta situação, se $p > 3$ é um número primo qualquer, então $p \neq A$. Portanto, se $p = B$ ou $p = C$, necessariamente, devemos ter $p = 6m - 1$ ou $p = 6m + 1$ para algum $m > 1$.

Caso B e C são ambos primos, então são ditos **Primos Gêmeos**. Assim, do Lema 1, todo par de primos gêmeos (p_1, p_2) com $3 < p_1 < p_2$, as parcelas $p_1 - 2$ e $p_2 + 2$ são múltiplos de 3. Com isto, fica evidente que a terna (3,5,7) é a única existente de primos trigêmeos.

Estratégia 4 – Aplicar a verificação imediata de primalidade do fator q no produto $p \cdot q$, sendo p um primo que satisfaz a condição do seguinte lema:

Lema 2 Se $p > \sqrt[3]{n}$ é o menor fator primo do número composto n, então $\frac{n}{p}$ também é primo.

Demonstração

Suponha $p > \sqrt[3]{n}$ e $n = p \cdot m$, então $p^3 > n$, isto é, $p^2 > \frac{n}{p} = m$. Logo, se $q \neq m$ é o menor fator primo de $m > 1$, seguiria que $q < \sqrt{m} < p$. Mas, isto contradiz a hipótese de que p é o menor fator primo de n, pois, $q|n$. Portanto, $q = m$, ou seja, $m = \frac{n}{p}$ é o outro fator primo de n. E ainda, $\frac{n}{p} > \frac{n}{\sqrt{n}} \Rightarrow m > \sqrt{n}$. □□□

EXEMPLO 1 Para ilustrar a aplicação deste Lema 2, considere o seguinte **número de Euclides** (SAUTOY, 2007) dado pelo produto dos 13 primeiros números primos acrescido de 1, ou seja,

$$n_{13} = 13^{\#} + 1 = \left(\prod_{p \leq 13} p \right) + 1 = 30.031$$

O algoritmo identifica o menor fator primo de n_{13} como sendo $p = 59$. Uma vez que $59 > \sqrt[3]{30.031} \approx 31{,}2$ é possível afirmar que $q = \frac{30.031}{59} = 509$ é o outro fator primo de n_{13}. Assim, segue de imediato a seguinte decomposição em fatores primos de $30.031 = 59 \cdot 509$.

EXEMPLO 2 Para ilustrar da utilização conjunta das quatro estratégias acima, considere os **Números de Fermat** como sendo inteiros positivos na forma $F_j = 2^{2^j} + 1$ para $j \in \mathbb{N}$. Assim:

$$F_0 = 2^{2^0} + 1 = 3, \qquad F_1 = 2^{2^1} + 1 = 5, \qquad F_2 = 2^{2^2} + 1 = 17, \qquad F_3 = 2^{2^3} + 1 = 257,$$

$F_4 = 2^{16} + 1 = 65.537$ e o número de 10 dígitos $F_5 = 2^{32} + 1 = 4.294.967.297$

Para efetuar, simultaneamente, um teste de primalidade nestes seis primeiros números de Fermat, aplicou-se o proposto algoritmo de decomposição no produto

$$N = \prod_{j=0}^{5} \left(2^{2^j} + 1 \right) = 18.446.744.073.709.551.615$$

e obteve-se $N = 3 \cdot 5 \cdot 17 \cdot 257 \cdot 641 \cdot 65.537 \cdot 6.700.417 = F_0 \cdot F_1 \cdot F_2 \cdot F_3 \cdot 641 \cdot F_4 \cdot 6.700.417$, assim, $F_5 = 641 \cdot 6.700.417$, ou seja, F_5 é um número de Fermat que não é primo.

O console de saída do algoritmo de decomposição apresentou o seguinte formato:

```
SEARCHING p IN 5 .. 2479700524 AND 1 ITERATION(S) TO FIND THE PRIME 5
SEARCHING p IN 5 .. 1108955787 AND 5 ITERATION(S) TO FIND THE PRIME 17
SEARCHING p IN 17 .. 268961285 AND 81 ITERATION(S) TO FIND THE PRIME 257
SEARCHING p IN 257 .. 16777344 AND 129 ITERATION(S) TO FIND THE PRIME 641
SEARCHING p IN 641 .. 662665 AND 21633 ITERATION(S) TO FIND THE PRIME 65537

THE LARGEST PRIME BY Lema2 IS 6700417

DECOMPOSITION OF NUMBER 18.446.744.073.709.551.615

[3, 5, 17, 257, 641, 65537, 6700417]

time for verification: 0.035 seconds.
```

Nesta decomposição, o fator primo $k_0 = 3$ foi logo identificado na estratégia 1.

Com o uso das estratégias 2 e 3 foram identificados:

- no intervalo $5 \leq k_1 \leq \sqrt{\frac{N}{k_0}}$ o fator primo $k_1 = 5$ após 1 iteração;
- no intervalo $k_1 \leq k_2 \leq \sqrt{\frac{N}{k_0 \cdot k_1}}$ o fator primo $k_2 = 17$ após 5 iterações;
- no intervalo $k_2 \leq k_3 \leq \sqrt{\frac{N}{k_0 \cdot k_1 \cdot k_2}}$ o fator primo $k_3 = 257$ após 81 iterações;
- no intervalo $k_3 \leq k_4 \leq \sqrt{\frac{N}{k_0 \cdot k_1 \cdot k_2 \cdot k_3}}$ o fator primo $k_4 = 641$ após 129 iterações;
- no intervalo $k_4 \leq k_5 \leq \sqrt{\frac{N}{k_0 \cdot k_1 \cdot k_2 \cdot k_3 \cdot k_4}}$ o fator primo $k_5 = 65.537$ após 21.633 iterações.

O maior primo divisor de N, ou seja, o fator $k_6 = \frac{N}{k_0 \cdot k_1 \cdot k_2 \cdot k_3 \cdot k_4 \cdot k_5} = 6.700.417$, foi automaticamente identificado pela estratégia 4 devido o fato de que k_5 satisfaz o Lema 2, ou seja, $\sqrt[3]{k_5 \cdot k_6} < k_5$.

O Algoritmo de Decomposição

O código de decomposição em fatores primos, baseados nas estratégias da seção anterior e no critério de divisão dado em (2), foi implementado em linguagem Python e segue disponível no *script*:

```
1  import math
2  from time import time; time0=time()
3
4  N = int(427624854937651133332883063661943206945693317653048683060135409348416)
5
6  number=N; fatores=[]
7
8  while (N%2==0):
9    fatores.append(2)
10   N=N//2
11
12 while (N%3==0):
13   fatores.append(3)
14   N=N//3
15
16 fator=N; aux=int(5)
17
18 def fatora(valor):
19   n = valor*valor; itera=0
20   nMax=int(math.pow(valor,.5))
21   for primo in range(aux,nMax+1, 2):
22        if ((primo-1)%6==0) or ((primo+1)%6==0):
23          num=(n-primo*primo)//2
24          itera=itera +1
25              if (num%primo==0):
26                  print("SEARCHING p IN",aux,"..",nMax+1,"AND",itera,"ITERATION(S) TO FIND THE PRIME", primo)
27             return primo
28    return valor
29
30 while (fator!=1):
31   fator2=fatora(fator)
32   fatores.append(fator2)
33   fator3=math.pow(fator,1/3)
34   if (fator2<=fator3) or (fator==fator2):
35     aux=(fator2-1)//2
36     fator=(fator//fator2)
37   else:
38     fatores.append(fator//fator2)
39     print(); print("PRIME BY Lema2",fator//fator2)
40     fator=1
41
42 print("DECOMPOSITION OF NUMBER", number)
43 print(fatores)
44 time1=time(); print("time for verification:", 1.5*(time1-time0),"seconds.")
```

CONSOLE 1/A SPYDER

```
Python 3.7.1 (default, Dec 10 2018, 22:54:23) [MSC v.1915 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.2.0 -- An enhanced Interactive Python.
runfile('C:/Users/Alessandro/Documents/PITAGORAS/FatoraN.py', wdir='C:/Users/Alessandro/Documents/PITAGORAS')

SEARCHING p IN 5 .. 16582054803249664894434197215641 7 AND 9 ITERATION(S) TO FIND THE PRIME 29
SEARCHING p IN 29 .. 307921027435985380685483973017 61 AND 1 ITERATION(S) TO FIND THE PRIME 29
SEARCHING p IN 29 .. 57179499321550572095669356462 09 AND 1 ITERATION(S) TO FIND THE PRIME 29
SEARCHING p IN 29 .. 10617966463309839791219351224 33 AND 1 ITERATION(S) TO FIND THE PRIME 29
SEARCHING p IN 29 .. 19717068731569160850403386982 5 AND 1 ITERATION(S) TO FIND THE PRIME 29
SEARCHING p IN 29 .. 36613677459689105761478115329 AND 174 ITERATION(S) TO FIND THE PRIME 547
SEARCHING p IN 547 .. 15654878027478749666124759 05 AND 1 ITERATION(S) TO FIND THE PRIME 547
SEARCHING p IN 547 .. 669354249720093351839006 73 AND 1 ITERATION(S) TO FIND THE PRIME 547
SEARCHING p IN 547 .. 28619521074001369209241 61 AND 1 ITERATION(S) TO FIND THE PRIME 547
SEARCHING p IN 547 .. 1223682357806386223513 61 AND 2461 ITERATION(S) TO FIND THE PRIME 7927
SEARCHING p IN 7927 .. 1374403554390935863297 AND 1 ITERATION(S) TO FIND THE PRIME 7927
SEARCHING p IN 7927 .. 15436891103902943233 AND 1 ITERATION(S) TO FIND THE PRIME 7927
SEARCHING p IN 7927 .. 173382560160329985 AND 32273 ITERATION(S) TO FIND THE PRIME 104743
SEARCHING p IN 104743 .. 535726225684923 AND 1 ITERATION(S) TO FIND THE PRIME 104743
SEARCHING p IN 104743 .. 1655314055931 AND 1 ITERATION(S) TO FIND THE PRIME 104743
SEARCHING p IN 104743 .. 5114673303 AND 398327 ITERATION(S) TO FIND THE PRIME 1299721
SEARCHING p IN 1299721 .. 4486347 AND 1 ITERATION(S) TO FIND THE PRIME 1299721

THE LARGEST PRIME BY Lema2 15485867

DECOMPOSITION OF NUMBER 427.624.854.937.651.133.332.883.063.661.943.206.945.693.317.653.048.683.060.135.409.348.416

[2, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 29, 29, 29, 29, 29, 547, 547, 547, 547, 7927, 7927, 7927, 104743, 104743, 104743, 1299721, 1299721, 15485867]

time for verification: 0.59 seconds.
```

Uma vantagem neste código que efetua operações para investigar valores ímpares de $k < a$, tais que $k|\left(\frac{a^2-k^2}{2}\right)$ quando a é um número composto, pode ser citada como a disponibilização de todas as trincas pitagóricas na forma $(a, b, b+k)$. Além da decomposição $F_5 = 641 \cdot 6.700.417$, o algoritmo ainda identifica as trincas pitagóricas com os valores de $k \in \{1, 641, 641^2, 6700417\}$. Estas únicas trincas são:

$$\begin{cases} (F_5, 9223372041149743104, 9223372041149743105); k = 1 \\ (F_5, 1438903594563 1104, 1438903594563 1745); k = 641 \\ (F_5, 22447793781504, 22447794192385); k = 641^2 \\ (F_5, 1376533668480, 1376540368897); k = F_5/641 \end{cases}$$

Da primeira trinca pitagórica acima, segue que

$$F_5^2 = 9.223.372.041.149.743.104 + 9.223.372.041.149.743.105$$

Isto é válido sempre que $k = 1$, pois $a^2 = c^2 - b^2 = (c+b)\underbrace{(c-b)}_{k=1} = b + c$

RESULTADOS

Nesta seção, o algoritmo de decomposição baseado na unicidade das trincas pitagóricas será aplicado na verificação de primalidade dos famosos números de Mersenne e, também, na identificação dos menos conhecidos, os Primos Hipotenusas. Um teste de verificação de primalidade também foi aplicado numa conhecida sequência de grandes primos em progressão aritmética para avaliar a confiança nos resultados do código proposto.

Primos de Mersenne

Segundo Sautoy (2015), Mersenne (1588-1648) percebeu que os números $M_n = 2^n - 1$ poderiam ser primos caso n também fosse um número primo. No entanto, aplicando o código de decomposição no produto $(2^{11} - 1) \cdot (2^{23} - 1) = 17.171.478.529$ determina-se:

```
SEARCHING p IN 5 .. 131039 AND 7 ITERATION(S) TO FIND THE PRIME 23
SEARCHING p IN 23 .. 27323 AND 9 ITERATION(S) TO FIND THE PRIME 47
SEARCHING p IN 47 .. 3985 AND 15 ITERATION(S) TO FIND THE PRIME 89
SEARCHING p IN 89 .. 178481 AND 112 ITERATION(S) TO FIND THE PRIME 178481

DECOMPOSITION OF NUMBER 17.171.478.529

[23, 47, 89, 178481]

time for verification: 0.0 seconds.
```

ou seja, M_{11} e M_{23} são ambos números compostos.

Mersenne afirmou, posteriormente, que para os valores de n dados por $2, 3, 5, 7, 19, 31, 67, 127$ e 257, M_n resultaria em um número primo. Em 1876 e sem auxílio de calculadoras mecânicas, Édouard Lucas (1842-1891) demonstrou que M_{127}, um número de 39 dígitos, era, de fato, um número primo[4]. Após elaborar um teste conhecido por Lucas-Lehmer (MARTINEZ, et al. 2015), foi

[4] Um primo record que permaneceu 75 anos até a descoberta, em 1951, de um número primo maior. https://en.wikipedia.org/wiki/Largest_known_prime_number

demostrado que M_{257} é um número composto. E ainda, de acordo com Lucas, foi possível afirmar que a única trinca pitagórica contendo o primo M_{127} é dada por:

$$(a = M_{127}\,, b = (M_{127} + 1) \cdot M_{126}\,, c = b + 1)$$

A verificação de primalidade dos números de Mersenne dados por: $M_2 = 3$, $M_3 = 7$, $M_5 = 31$, $M_7 = 127$, $M_{19} = 524.287$, $M_{31} = 2.147.483.647$, $M_{67} = 147.573.952.589.676.412.927$ pode ser realizada, simultaneamente, aplicando o código de decomposição em um número N que é obtido pela multiplicação destes sete fatores.

O console de saída fornece:

```
SEARCHING p IN 5 .. 67668423626175668224 AND 2 ITERATION(S) TO FIND THE PRIME 7
SEARCHING p IN 7 .. 2557626007523262464O AND 9 ITERATION(S) TO FIND THE PRIME 31
SEARCHING p IN 31 .. 4593631915527500288 AND 33 ITERATION(S) TO FIND THE PRIME 127
SEARCHING p IN 127 .. 407618918210493888 AND 174721 ITERATION(S) TO FIND THE PRIME 524287
SEARCHING p IN 524287 .. 562949953290240 AND 64394479 ITERATION(S) TO FIND THE PRIME 193707721
SEARCHING p IN 193707721 .. 40447931951 AND 651258643 ITERATION(S) TO FIND THE PRIME 2147483647

THE LARGEST PRIME BY Lema2 IS 761838257287

DECOMPOSITION OF NUMBER 13.737.046.668.154.706.597.306.706.000.546.963.569.931

[3, 7, 31, 127, 524287, 193707721, 2147483647, 761838257287]

time for verification: 1263.95 seconds.
```

logo, o código indica $N = M_2 \cdot M_3 \cdot M_5 \cdot M_7 \cdot M_{19} \cdot 193707721 \cdot M_{31} \cdot 761838257287$, assim, $M_{67} = 193707721 \cdot 761838257287$ é outro número de Mersenne que não é primo[5].

No ano de 1732, o brilhante matemático Leonhard Euler (1707-1783) equivocou-se ao afirmar que M_{41} e M_{47} eram ambos números primos (HARDY e WRIGHT, 1960). Respectivamente em 1883 e 1886, e de forma independente, Pervusin e Seelhoff acrescentaram na lista dos números primos de Mersenne o $M_{61} = 2.305.843.009.213.693.951$ (RASSIAS, 2010). Após poucos minutos de esforço computacional, comprova-se, simultaneamente, que M_{41} e M_{47} são números compostos e que M_{61} é primo. Para isto, é realizado a decomposição de $N = (2^{41} - 1) \cdot (2^{47} - 1) \cdot (2^{61} - 1)$ para demonstrar estes fatos.

```
SEARCHING p IN 5 .. 26713738906275369975808 AND 783 ITERATION(S) TO FIND THE PRIME 2351
SEARCHING p IN 2351 .. 550945147594834968576 AND 722 ITERATION(S) TO FIND THE PRIME 4513
SEARCHING p IN 4513 .. 8201167757968887808 AND 2952 ITERATION(S) TO FIND THE PRIME 13367
SEARCHING p IN 13367 .. 70934697595644928 AND 4417055 ITERATION(S) TO FIND THE PRIME 13264529
SEARCHING p IN 13264529 .. 19476584743001 AND 50415609 ITERATION(S) TO FIND THE PRIME 164511353
SEARCHING p IN 164511353 .. 2305843009213693951 AND 451329633 ITERATION(S) TO FIND THE PRIME 2305843009213693951

DECOMPOSITION OF NUMBER 713.623.846.352.650.351.063.598.637.564.999.302.358.499.327

[2351, 4513, 13367, 13264529, 164511353, 2305843009213693951]

time for verification: 758.56 seconds.
```

O que resulta na decomposição $N = 2351 \cdot 4513 \cdot 13367 \cdot 13264529 \cdot 164511353 \cdot M_{61}$

Assim, o código foi útil para verificar que M_{61} é, de fato, um número primo de Mersenne e, que M_{41} e M_{47} são ambos números compostos.

[5] Em 1903, Frank Cole (1861-1926) apresentou a decomposição do número M_{67} após três anos de cálculos.

Primos Hipotenusas

Uma curiosa associação entre os números primos $p > 2$ e as trincas (a, b, c) são os *Triângulos Primos de Pitágoras* definidos por $(a = p, b, c = q)$, sendo q outro número primo. As trincas $(3,4,5), (5,12,13)$ e $(11,60,61)$ são exemplos de triângulo primos de Pitágoras.

Algumas propriedades interessantes podem ser demonstradas nestas trincas $(a, b, b + k)$:

1.) Sendo $a = p$ um número primo, então $k = 1$, ou seja, $c = b + 1$.

2.) O cateto maior b é um número par. Para isto, em (2) faça $b = \frac{p^2-1}{2} = \frac{1}{2}\underbrace{(p-1)}_{par}\underbrace{(p+1)}_{par}$.

3.) Se $a > 5$ e $c = b + 1 = \frac{a^2+1}{2}$ são ambos números primos, então o cateto $a \equiv \pm 1 \bmod 10$ e a hipotenusa $c \equiv 1 \bmod 10$. No caso em que o cateto primo satisfaz $a \equiv \pm 3 \bmod 10$, verifica-se que $c \equiv 5 \bmod 10$, que não é primo[6].

Apesar dessas restrições, conjecturam-se a existência de infinitos triângulos primos de Pitágoras (RIBENBOIM, 2015).

Considere p primo e $q_p = \frac{p^2+1}{2}$. Se q_p também for primo, dizemos que q_p é um **Primo Hipotenusa**. Desta forma, toda trinca $\left(p, \frac{p^2-1}{2}, q_p\right)$ determina um triângulo primo de Pitágoras.

Além dos já citados: $5, 13$ e 61, o código de decomposição permitiu a revelação dos próximos Primos Hipotenusas como sendo: $q_{19} = 181$, $q_{29} = 421$, $q_{59} = 1741$, $q_{61} = 1861$ e $q_{71} = 2521$. De fato, decompondo o número obtido da multiplicação dos 10 ímpares envolvidos, temos 10 fatores primos:

```
SEARCHING p IN 5 .. 296029660727 AND 6 ITERATION(S) TO FIND THE PRIME 19
SEARCHING p IN 19 .. 67913861863 AND 4 ITERATION(S) TO FIND THE PRIME 29
SEARCHING p IN 29 .. 12611287545 AND 11 ITERATION(S) TO FIND THE PRIME 59
SEARCHING p IN 59 .. 1641849791 AND 2 ITERATION(S) TO FIND THE PRIME 61
SEARCHING p IN 61 .. 210217324 AND 4 ITERATION(S) TO FIND THE PRIME 71
SEARCHING p IN 71 .. 24948206 AND 38 ITERATION(S) TO FIND THE PRIME 181
SEARCHING p IN 181 .. 1854385 AND 81 ITERATION(S) TO FIND THE PRIME 421
SEARCHING p IN 421 .. 90377 AND 441 ITERATION(S) TO FIND THE PRIME 1741
SEARCHING p IN 1741 .. 2166 AND 41 ITERATION(S) TO FIND THE PRIME 1861

THE LARGEST PRIME BY Lema2 IS 2521

DECOMPOSITION OF NUMBER 87.633.560.030.293.446.279.359

[19, 29, 59, 61, 71, 181, 421, 1741, 1861, 2521]

time for verification: 0.01 seconds.
```

Os respectivos triângulos primos de Pitágoras da sequência acima são: $(19,180,181)$, $(29,420,421)$, $(59,1740,1741)$, $(61,1860,1861)$ e $(71,2520,2521)$.

A penúltima trinca é curiosa pois o cateto $a = 61$, primo gêmeo do cateto da trinca anterior $a = 59$, também é um Primo Hipotenusa na trinca $(11,60,61)$[7].

[6] É possível afirmar que, se a for ímpar, então não existe trinca pitagórica (a, b, c) que satisfaça $c \equiv 7 \bmod 10$.
[7] 181 e 1741 também são exemplos de Primos Hipotenusa que definem, respectivamente, outros Primos Hipotenusas: 16.381 e 1.515.541.

Na fatoração de $N = \frac{1439^2+1}{2} \cdot \frac{1459^2+1}{2} \cdot \frac{1489^2+1}{2} \cdot \frac{1499^2+1}{2} \cdot \frac{1531^2+1}{2}$, dada pelo código por:

```
SEARCHING p IN 5 .. 1268274066259589 AND 345120 ITERATION(S) TO FIND THE PRIME 1035361
SEARCHING p IN 1035361 .. 1246428041850 AND 9661 ITERATION(S) TO FIND THE PRIME 1064341
SEARCHING p IN 1064341 .. 1208166571 AND 14741 ITERATION(S) TO FIND THE PRIME 1108561
SEARCHING p IN 1108561 .. 1147485 AND 4981 ITERATION(S) TO FIND THE PRIME 1123501

THE LARGEST PRIME BY Lema2 IS 1171981

DECOMPOSITION OF NUMBER 1.608.519.107.146.632.351.098.878.568.941

[1035361, 1064341, 1108561, 1123501, 1171981]

time for verification: 0.57 seconds.
```

é obtido, acima de 1.000.000, os seguintes Primos Hipotenusas: 1.035.361, 1.064.341, 1.108.561, 1.123.501 e 1.171.981.

A verificação de que $1439, 1459, 1489, 1499$ e 1531 são todos primos é obtida pela decomposição:

```
SEARCHING p IN 5 .. 84702027 AND 479 ITERATION(S) TO FIND THE PRIME 1439
SEARCHING p IN 1439 .. 2232869 AND 8 ITERATION(S) TO FIND THE PRIME 1459
SEARCHING p IN 1459 .. 58456 AND 11 ITERATION(S) TO FIND THE PRIME 1489
SEARCHING p IN 1489 .. 1514 AND 4 ITERATION(S) TO FIND THE PRIME 1499

THE LARGEST PRIME BY Lema2 IS 1531

DECOMPOSITION OF NUMBER 7.174.433.378.888.341

[1439, 1459, 1489, 1499, 1531]

time for verification: 0.00 seconds.
```

NOTA DOS AUTORES

Primos Pitagoreanos são primos na forma $q = 1 + 4n$, para algum $n \in \mathbb{N}$. Estes primos possuem esta denominação pois Ramanujan (1887-1920) demostrou ser possível realizar a decomposição $q = x^2 + y^2$ para valores únicos dos inteiros positivos x e $y < x$ (BERNDT, 1994). Outra característica importante é que todo Primo Pitagoreano q é a hipotenusa numa única trinca (a, b, q). Para isto, considere $a = x^2 - y^2$ e $b = 2xy$. E ainda, é possível afirmar que todo Primo Hipotenusa $q_p = \frac{p^2+1}{2} = \frac{(6m \pm 1)^2+1}{2} = 1 + 4N$, $N \in \mathbb{N}$, define um Primo Pitagoreano, no entanto, a recíproca não é verdadeira.

O **Teorema de Dirichlet** garante a existência de infinitos primos p em toda sequência do tipo $a + bn$, sendo $n \in \mathbb{N}$, a e b coprimos[8] (MARTINEZ, et al. 2015). Logo, existem infinitos primos da forma $q = 1 + 4n$, ou seja, entre outras muitas propriedades, o teorema de Dirichlet demonstra a existência de infinitas trincas pitagóricas (a, b, q) em que a hipotenusa é um número primo.

Primos em Progressões Aritméticas

Segundo Martinez, (2015), projetos cooperativos para investigação de grandes primos pela internet proporcionaram a descoberta de progressões aritméticas (PA) formadas exclusivamente por primos. Em 2015, Bryan Little apresentou a uma sequência de 26 primos b_n obtidos da seguinte PA:

[8] $mdc(a, b) = 1$, ou seja, a e b são primos entre si.

$$b_n = 161.004.359.399.459.161 + 47.715.109 \cdot 23^{\#} \cdot n \quad (5)$$

sendo $23^{\#} = \prod_{p \leq 23} p = 223.092.870$ o produto dos 23 primeiros números primos.

Para evidenciar que o presente algoritmo de decomposição baseado em trincas pitagóricas, produz respostas corretas, foi efetuado um teste de primalidade nos 26 números b_n da sequência de Bryan. Após realizar alguns ajustes no console de saída do código proposto, os resultados das avaliações seguem abaixo:

SEARCHING p IN 5 .. 161004359399459161 AND 133751161 ITERATION(S) TO FIND THE PRIME **b00 = 161.004.359.399.459.161**
time for verification: 131.95 seconds.

SEARCHING p IN 5 .. 171649260008631991 AND 138101918 ITERATION(S) TO FIND THE PRIME **b01 = 171.649.260.008.631.991**
time for verification: 191.87 seconds.

SEARCHING p IN 5 .. 182294160617804821 AND 142319733 ITERATION(S) TO FIND THE PRIME **b02 = 182.294.160.617.804.821**
time for verification: 142.98 seconds.

SEARCHING p IN 5 .. 192939061226977651 AND 146416096 ITERATION(S) TO FIND THE PRIME **b03 = 192.939.061.226.977.651**
time for verification: 197.47 seconds.

SEARCHING p IN 5 .. 203583961836150481 AND 150400930 ITERATION(S) TO FIND THE PRIME **b04 = 203.583.961.836.150.481**
time for verification: 149.91 seconds.

SEARCHING p IN 5 .. 214228862445323311 AND 154282878 ITERATION(S) TO FIND THE PRIME **b05 = 214.228.862.445.323.311**
time for verification: 215.61 seconds.

SEARCHING p IN 5 .. 224873763054496141 AND 158069520 ITERATION(S) TO FIND THE PRIME **b06 = 224.873.763.054.496.141**
time for verification: 155.39 seconds.

SEARCHING p IN 5 .. 235518663663668971 AND 161767550 ITERATION(S) TO FIND THE PRIME **b07 = 235.518.663.663.668.971**
time for verification: 208.29 seconds.

SEARCHING p IN 5 .. 246163564272841801 AND 165382910 ITERATION(S) TO FIND THE PRIME **b08 = 246.163.564.272.841.801**
time for verification: 162.89 seconds.

SEARCHING p IN 5 .. 256808464882014631 AND 168920909 ITERATION(S) TO FIND THE PRIME **b09 = 256.808.464.882.014.631**
time for verification: 237.39 seconds.

SEARCHING p IN 5 .. 267453365491187461 AND 172386311 ITERATION(S) TO FIND THE PRIME **b10 = 267.453.365.491.187.461**
time for verification: 171.85 seconds.

SEARCHING p IN 5 .. 278098266100360291 AND 175783409 ITERATION(S) TO FIND THE PRIME **b11 = 278.098.266.100.360.291**
time for verification: 229.63 seconds.

SEARCHING p IN 5 .. 288743166709533121 AND 179116090 ITERATION(S) TO FIND THE PRIME **b12 = 288.743.166.709.533.121**
time for verification: 176.96 seconds.

SEARCHING p IN 5 .. 299388067318705951 AND 182387884 ITERATION(S) TO FIND THE PRIME **b13 = 299.388.067.318.705.951**
time for verification: 233.52 seconds.

SEARCHING p IN 5 .. 310032967927878781 AND 185602013 ITERATION(S) TO FIND THE PRIME **b14 = 310.032.967.927.878.781**
time for verification: 184.42 seconds.

SEARCHING p IN 5 .. 320677868537051611 AND 188761420 ITERATION(S) TO FIND THE PRIME **b15 = 320.677.868.537.051.611**
time for verification: 247.75 seconds.

SEARCHING p IN 5 .. 331322769146224441 AND 191868811 ITERATION(S) TO FIND THE PRIME **b16 = 331.322.769.146.224.441**
time for verification: 190.46 seconds.

SEARCHING p IN 5 .. 341967669755397271 AND 194926672 ITERATION(S) TO FIND THE PRIME **b17 = 341.967.669.755.397.271**
time for verification: 260.43 seconds.

SEARCHING p IN 5 .. 352612570364570101 AND 197937298 ITERATION(S) TO FIND THE PRIME **b18 = 352.612.570.364.570.101**
time for verification: 196.83 seconds.

SEARCHING p IN 5 .. 363257470973742931 AND 200902814 ITERATION(S) TO FIND THE PRIME **b19 = 363.257.470.973.742.931**
time for verification: 255.67 seconds.

SEARCHING p IN 5 .. 373902371582915761 AND 203825189 ITERATION(S) TO FIND THE PRIME **b20 = 373.902.371.582.915.761**
time for verification: 200.66 seconds.

SEARCHING p IN 5 .. 384547272192088591 AND 206706251 ITERATION(S) TO FIND THE PRIME **b21 = 384.547.272.192.088.591**
time for verification: 259.74 seconds.

SEARCHING p IN 5 .. 395192172801261421 AND 209547706 ITERATION(S) TO FIND THE PRIME **b22 = 395.192.172.801.261.421**
time for verification: 209.57 seconds.

SEARCHING p IN 5 .. 405837073410434251 AND 212351142 ITERATION(S) TO FIND THE PRIME **b23 = 405.837.073.410.434.251**

time for verification: 257.89 seconds.

SEARCHING p IN 5 .. 416481974019607081 AND 215118048 ITERATION(S) TO FIND THE PRIME **b24 = 416.481.974.019.607.081**
time for verification: 221.71 seconds.

SEARCHING p IN 5 .. 427126874628779911 AND 217849814 ITERATION(S) TO FIND THE PRIME **b25 = 427.126.874.628.779.911**
time for verification: 225.97 seconds.

Assim, em tempo de execução razoável, o código proposto foi capaz de confirmar a primalidade dos 26 números apresentados na progressão aritmética de Bryan.

CONSIDERAÇÕES FINAIS

De característica didática e sem pretensões de otimização da complexidade computacional dos testes de primalidade, o algoritmo proposto relaciona termos relevantes no fascinante mundo dos números primos e, quando possível, insere algumas terminologias e contribuições de personagens ilustres que contribuíram no desenvolvimento da Teoria dos Números. O Lema 2 deste artigo é um artifício de considerável aplicabilidade para efetuar a decomposição de números compostos que são formados pelo produto de dois números primos contendo vários dígitos cada, como por exemplo, os *semiprimos* conhecidos por números RSA. Em relação aos números de Mersenne, os primos M_{89} e M_{107} foram posteriormente acrescidos em sua lista de primos.

A partir dos valores 5 e 13, todos os demais Primos Hipotenusas q_p, infinitos ou não, terminam com o dígito 1. E ainda, sendo $q_p = 1 + 4N$, o código permitiu verificar, através de simulações efetuadas, que $N \equiv 0 \bmod 15$, ou seja, se a trinca $(p > 5, b, q_p)$ for um Triângulo Primo de Pitágoras, então o cateto maior b será um múltiplo de 60 e $q_p = 1 + 60n$ para algum $n \in \mathbb{N}$, pertencerá a uma subsequência de $1 + 4N$ dos Primos Pitagoreanos.

Verificou-se, por fim, que a velocidade de execução do código de decomposição é proporcional à quantidade de fatores do número composto e não, necessariamente, da quantidade de seus dígitos. Outro fator que influencia a eficiência do teste de primalidade é o recurso computacional utilizado. Isto explica a alternância do tempo de execução observado na PA de Bryan, pois, enquanto os b_n de índices ímpares foram executado em computador pessoal, os de índices pares utilizaram eficácia da plataforma ***Colaboratory*** do Jupyter Notebook (https://colab.research.google.com/notebooks/intro.ipynb).

Estes resultados surpreendentes de primalidade, apoiados em propriedades básicas que sustentam a unicidade das trincas pitagóricas (a, b, c), é uma pequena contribuição que visa enriquecer a robusta área que estuda os números inteiros, ou seja, o palco comum que entrelaça matemáticos profissionais e entusiastas amadores através da fascinante e misteriosa Teoria dos Números.

Referências

BERNDT, B. C. **Ramanujan's Notebook Part IV**. New York: Springer-Verlag, 1994.

FIRMIANO, A.; SANTOS, J. P. M.; ELOY, M. E.; CARDOSO, C. E. As Infinitas Trincas Pitagóricas de Euclides **Revista Eletrônica Paulista CQD**, v. 17, p. 13-26, 2020.

HARDY, G. H.; WRIGHT, E. W. **Problem-Solving and Selected Topics in Number Theory:** in the Spirit of the Mathematical Olympiads. London: Oxford University Press, 1960.

MARTINEZ, F. B.; MOREIRA C. A.; SALDANHA, N.; TENGAN, E. **Teoria dos Números:** um passeio com primos e outros números familiares pelo mundo inteiro. 4. ed. Rio de Janeiro: Projeto Euclides, 2015.

RIBENBOIM, P. **Números Primos, amigos que causam problemas:** um Triálogo com o Papa Paulo. Rio de Janeiro: SBM, 2015.

RASSIAS, M. **Problem-Solving and Selected Topics in Number Theory:** in the Spirit of the Mathematical Olympiads. New York: Springer, 2010.

SAUTOY, M. **A Música dos Números Primos:** a história de um problema não resolvido na matemática. Rio de Janeiro: Zahar, 2007.