teoria de números computacional

cláudia mendes araújo

2024/2025

lcc+lmat | uminho

Consideremos a seguinte sequência de números racionais:

$$\frac{1}{1}, \frac{3}{2}, \frac{7}{5}, \frac{17}{12}, \frac{41}{29}, \frac{99}{70}, \ldots.$$

Depois de começar com $\frac{1}{1}$, a fração seguinte é obtida somando o numerador e o denominador para obter o novo denominador, e somando o anterior e o novo denominadores para obter o numerador.

Comparem-se este valores com a raiz quadrada de 2.

$$\frac{3}{2} - \sqrt{2} = 0,0857864376...$$
$$\frac{7}{5} - \sqrt{2} = -0,0142135623...$$
$$\frac{17}{12} - \sqrt{2} = 0,0024531042...$$
$$\frac{41}{29} - \sqrt{2} = -0,0004204589...$$
$$\frac{99}{70} - \sqrt{2} = 0,0000721519...$$
$$\frac{239}{169} - \sqrt{2} = -0.0000123789...$$
$$\frac{577}{408} - \sqrt{2} = 0,0000021239...$$
$$\frac{1393}{985} - \sqrt{2} = -0,0000003644...$$

Não são apenas boas aproximações, mas sim **aproximações surpreendentemente boas**.

A aproximação mais familiar para $\sqrt{2}$, $\frac{141}{100}$, tem um erro não superior a $\frac{1}{200}$.

Mas o erro para uma fração da sequência referida anteriormente é **menor que o inverso do dobro do quadrado do denominador**: por exemplo, $\frac{99}{70} - \sqrt{2} =$ $= 0,000072\ldots$, enquanto que $\frac{1}{2\times 70^2} = 0,00010\ldots$.

Este é um **algoritmo antigo**, de Teoria de Números, conhecido pelos gregos da era clássica.

A **Teoria de Números**, o estudo das propriedades dos inteiros, há muito que é indissociável dos algoritmos - procedimentos precisamente especificados que produzem um resultado desejado. Para encontrar algoritmos numéricos eficientes, devemos compreender a estrutura dos inteiros. Para explorar essa estrutura, somos auxiliados pelos algoritmos que já conhecemos.

A fatorização de inteiros grandes ilustra bem a necessidade de compreender a estrutura dos inteiros. Números inteiros com **100 dígitos** podem ser fatorizados de forma rotineira.

Poderia parecer que este é um problema para o qual uma abordagem ingénua, aliada a um grande poder computacional, funcionaria bem: basta gerar uma lista de números primos e realizar divisões sucessivas até encontrar um divisor primo.

Como veremos, gerar os possíveis divisores primos é fácil. O problema é que há demasiados. Se **o menor divisor primo do nosso número tiver 50 dígitos** (e, hoje em dia, isso nem é considerado especialmente grande), o Teorema dos Números Primos indica que **há cerca de $8,7 \times 10^{47}$ primos a testar** antes de chegarmos a esse divisor. Isso implica um número astronómico de divisões!

Imaginemos um processador ideal capaz de realizar um bilião (10^{12}) de divisões por segundo, e um milhão desses processadores a trabalhar em paralelo, cada um testando um subconjunto diferente de divisores possíveis.

Mesmo assim, **conseguiríamos testar apenas 10^{18} primos por segundo**. Parece muito, mas não é. Como há cerca de $3,2 \times 10^7$ segundos num ano, a fatorização levaria mais de **10^{22} anos** para ser concluída. O nosso universo tem menos de 2×10^{10} anos de idade.

De facto, não há outro problema em teoria dos números que desafie tanto a nossa compreensão dos inteiros como o **problema da fatorização**.

o algoritmo de Euclides estendido

teorema. (algoritmo da divisão) Dados dois números inteiros a e b tais que $b > 0$ existe um e um só inteiro q e existe um e um só inteiro r tais que

$$a = bq + r \quad \text{e} \quad 0 \leq r < b.$$

demonstração. existência:

Consideremos o conjunto

$$S = \{a - xb \in \mathbb{N}_0 : x \in \mathbb{Z}\}.$$

Se $0 \in S$, então 0 é o elemento mínimo de S. Se $0 \notin S$, então $S \subseteq \mathbb{N}$. Temos

$$\begin{aligned} b \geq 1 \quad &\Rightarrow |a|b \geq |a| \\ &\Rightarrow a + |a|b \geq a + |a| \geq 0 \\ &\Rightarrow a - (-|a|)b \geq 0. \end{aligned}$$

Como $-|a| \in \mathbb{Z}$, $a - (-|a|)b \in S$ e, portanto, $S \neq \varnothing$.

Pelo Princípio da Boa Ordenação de $\mathbb{N}$ (todo o subconjunto não vazio de $\mathbb{N}$ tem elemento mínimo), existe o elemento mínimo de S. Seja $r = \min S$.

Então, existe $q \in \mathbb{Z}$ tal que $r = a - qb$ e $r \geq 0$. Equivalentemente, existe $q \in \mathbb{Z}$ tal que

$$a = qb + r \text{ e } r \geq 0.$$

Suponhamos agora que $b \leq r$. Então,

$$a - (q+1)b = a - qb - b = r - b \geq 0,$$

pelo que $a - (q+1)b \in S$.

Assim,

$$r - b \in S$$

e

$$r - b \geq \min S = r,$$

o que é um absurdo, pois $b > 0$. O absurdo resultou de termos suposto que $b \leq r$.

Logo, $r < b$.

unicidade:

Sejam $q, q', r, r' \in \mathbb{Z}$ tais que

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

Por um lado, $b(q - q') = r' - r$ e, portanto,

$$b|q - q'| = |r' - r|. \qquad (*)$$

Por outro lado,

$$\begin{aligned} \left\{ \begin{array}{l} 0 \leq r < b \\ 0 \leq r' < b \end{array} \right. \quad &\Leftrightarrow \quad \left\{ \begin{array}{l} 0 \leq r' < b \\ -b < -r \leq 0 \end{array} \right. \\ &\Rightarrow \quad -b < r' - r < b \\ &\Leftrightarrow \quad |r' - r| < b. \end{aligned}$$

Logo, de $(*)$, temos que $b|q - q'| < b$, pelo que $0 \leq |q - q'| < 1$. Como $q - q' \in \mathbb{Z}$, concluímos que $q - q' = 0$, i.e., $q = q'$. Novamente de $(*)$, concluímos que $r = r'$.

corolário. Sejam $a, b \in \mathbb{Z}$ com $b \neq 0$. Então, existem inteiros q e r, univocamente determinados, tais que

$$a = bq + r \ \text{ e } \ 0 \leq r < |b|.$$

demonstração. Falta estudar o caso em que $b \in \mathbb{Z}^-$.

Como $|b| > 0$, aplicando o teorema anterior, temos que existem um e um só $q' \in \mathbb{Z}$ e um e um só $r' \in \mathbb{Z}$ tais que

$$a = q'|b| + r' \text{ e } 0 \leq r' < |b|.$$

Como $|b| = -b$, obtemos

$$a = (-q')b + r' \text{ e } 0 \leq r' < |b|,$$

o que prova o resultado pretendido.

definição. Sejam $a, b \in \mathbb{Z}$, $a \neq 0$ ou $b \neq 0$. Chama-se máximo divisor comum de a e b, e representa-se por $\mathrm{m.d.c.}(a, b)$, ao inteiro positivo d tal que:

(i) $d \mid a$ e $d \mid b$;

(ii) Para todo $c \in \mathbb{N}$, $(c \mid a \text{ e } c \mid b) \Rightarrow c \leq d$.

Por outras palavras,

$$\mathrm{m.d.c.}(a, b) = \max\{d \in \mathbb{Z} : d \mid a \wedge d \mid b\}.$$

Se $a = b = 0$, $\mathrm{m.d.c.}(a, b) = 0$.

obs. $\mathrm{m.d.c.}(a, 0) = a$, para todo o inteiro a.

lema. Para quaisquer $a, b \in \mathbb{Z}$,

$$\mathrm{m.d.c.}(a, b) = \mathrm{m.d.c.}(b, a) = \mathrm{m.d.c.}(\pm a, \pm b) = \mathrm{m.d.c.}(a, b - a) = \mathrm{m.d.c.}(a, b + a).$$

lema. Para quaisquer $a, b, n \in \mathbb{Z}$, $\mathrm{m.d.c.}(a, b) = \mathrm{m.d.c.}(a, b - na)$.

lema. Sejam a e b inteiros não nulos e $q, r \in \mathbb{Z}$ tais que $a = qb + r$ e $0 \leq r < b$. Então, $\mathrm{m.d.c.}(a, b) = \mathrm{m.d.c.}(b, r)$.

exemplo. Usemos o algoritmo da divisão repetidamente para determinar $\mathrm{m.d.c.}(2261, 1275)$.

Dividindo 2261 por 1275, obtemos que

$$2261 = 1 \times 1275 + 986.$$

Assim, $\mathrm{m.d.c.}(2261, 1275) = \mathrm{m.d.c.}(1275, 986)$.

Dividindo 1275 por 986, temos que

$$1275 = 1 \times 986 + 289.$$

Logo, $\mathrm{m.d.c.}(1275, 986) = \mathrm{m.d.c.}(986, 289)$.

Continuando:

$$986 = 3 \times 289 + 119,$$

$$289 = 2 \times 119 + 51,$$

$$119 = 2 \times 51 + 17$$

pelo que $\mathrm{m.d.c.}(2261, 1275) = \mathrm{m.d.c.}(1275, 986) = \mathrm{m.d.c.}(986, 289) =$
$= \mathrm{m.d.c.}(289, 119) = \mathrm{m.d.c.}(119, 51) = \mathrm{m.d.c.}(51, 17)$.

Dado que $17 \mid 51$, segue-se que $\mathrm{m.d.c.}(51, 17) = 17$, donde $\mathrm{m.d.c.}(2261, 1275) = 17$.

observação. Sejam $a, b \in \mathbb{Z}$ não simultaneamente nulos. Pretendemos determinar o $\mathrm{m.d.c.}(a, b)$ e escrevê-lo como combinação linear de a e de b. Como $\mathrm{m.d.c.}(|a|, |b|) = \mathrm{m.d.c.}(a, b) = \mathrm{m.d.c.}(b, a)$ e $\mathrm{m.d.c.}(a, 0) = a$, podemos estudar apenas o caso em que $a \geq b > 0$.

teorema. (algoritmo de Euclides) Sejam a e b inteiros tais que $a \geq b > 0$. Usando o algoritmo da divisão, considerem-se as sequências $r_0, r_1, \cdots, r_\ell, r_{\ell+1}$, $q_1, \cdots, q_\ell$ com $\ell \geq 1$ e

$$\begin{array}{lll}
a = r_0 & & \\
b = r_1 & & \\
r_0 = q_1 r_1 + r_2 & \text{onde} & 0 < r_2 < r_1 \\
r_1 = q_2 r_2 + r_3 & \text{onde} & 0 < r_3 < r_2 \\
\vdots & & \\
r_{i-1} = q_i r_i + r_{i+1} & \text{onde} & 0 < r_{i+1} < r_i \\
\vdots & & \\
r_{\ell-2} = q_{\ell-1} r_{\ell-1} + r_\ell & \text{onde} & 0 < r_\ell < r_{\ell-1} \\
r_{\ell-1} = q_\ell r_\ell + r_{\ell+1} & \text{onde} & r_{\ell+1} = 0.
\end{array}$$

então $\mathrm{m.d.c.}(a, b) = r_\ell$.

observações.

1. Ao calcular $\mathrm{m.d.c.}(a, b)$ a partir de $\mathrm{m.d.c.}(r_1 = b, r_2)$, baixamos a ordem de grandeza dos números (e assim sucessivamente);
2. Vamos obter, a certa altura, o resto 0, uma vez que a sequência decrescente de restos $a = r_0 \geq r_1 > r_2 > \cdots \geq 0$ não pode ter mais que a termos (uma vez que cada resto é um inteiro);
3. $\mathrm{m.d.c.}(a = r_0, b = r_1) = \mathrm{m.d.c.}(r_1, r_2) = \mathrm{m.d.c.}(r_2, r_3) = \cdots = \mathrm{m.d.c.}(r_{\ell-1}, r_\ell)$ $= \mathrm{m.d.c.}(r_\ell, r_{\ell+1}) = \mathrm{m.d.c.}(r_\ell, 0) = r_\ell$, ou seja, $\mathrm{m.d.c.}(a, b)$ é o último resto não nulo;
4. Temos ℓ passos no algoritmo de Euclides.

exemplo. Os passos usados no algoritmo de Euclides para determinar $\mathrm{m.d.c.}(252, 198)$ são:

$$\begin{aligned} 252 &= 1 \times 198 + 54, \\ 198 &= 3 \times 54 + 36, \\ 54 &= 1 \times 36 + 18, \\ 36 &= 2 \times 18. \end{aligned}$$

Podemos resumir estes passos numa tabela:

j	r_j	r_{j+1}	q_{j+1}	r_{j+2}
0	252	198	1	54
1	198	54	3	36
2	54	36	1	18
3	36	18	2	0

O último resto não nulo é 18, pelo que $\mathrm{m.d.c.}(252, 198) = 18$.

teorema. Sejam a e b inteiros tais que $a \geq b > 0$ e tais que, usando o algoritmo de Euclides, se obtêm as sequências de restos $r_0 \geq r_1 > \cdots > r_\ell > r_{\ell+1} = 0$ e quocientes $q_1, \cdots, q_\ell$ como descrito no teorema anterior. Então,

$$\ell \leq \frac{\log b}{\log \phi} + 1,$$

onde $\phi = \frac{1+\sqrt{5}}{2}$ é o número de ouro.

demonstração. Consideremos a sucessão de Fibonacci $(F_n)_n$.

Comecemos por notar que cada quociente $q_1, q_2, \cdots, q_{\ell-1}$ é maior ou igual a 1 e que $q_\ell \geq 2$, uma vez que $r_\ell < r_{\ell-1}$.

Assim,

$$\begin{gathered}
r_\ell \geq 1 = F_2 \\
r_{\ell-1} \geq 2r_\ell \geq 2F_2 = F_3 \\
r_{\ell-2} \geq r_{\ell-1} + r_\ell \geq F_3 + F_2 = F_4 \\
r_{\ell-3} \geq r_{\ell-2} + r_{\ell-1} \geq F_4 + F_3 = F_5 \\
\vdots \\
r_2 \geq r_3 + r_4 \geq F_{\ell-1} + F_{\ell-2} = F_\ell \\
r_1 \geq r_2 + r_3 \geq F_\ell + F_{\ell-1} = F_{\ell+1}.
\end{gathered}$$

Vimos, assim, que $b \geq F_{\ell+1}$.

Mostremos por indução que $F_{n+1} > \phi^{n-1}$, para o natural $n \geq 2$.

(I) $\underline{n = 2}$:

$\phi < 2 = F_3$.

Logo, $F_{2+1} > \phi^{2-1}$

$\underline{n = 3}$:

$\phi^2 = \frac{3+\sqrt{5}}{2} < 3 = F_4$.

Logo, $F_{3+1} > \phi^{3-1}$.

(II) Seja $n \in \mathbb{N}$ tal que $n \geq 3$ e admitamos que $\phi^{k-1} < F_{k+1}$ para todo o natural $k \in \{2, \cdots, n-1\}$.

É sabido que ϕ é uma solução da equação $x^2 - x - 1 = 0$, ou seja, $\phi^2 = \phi + 1$. Temos que

$$\begin{aligned} \phi^{n-1} &= \phi^{n-3}\phi^2 \\ &= \phi^{n-3}(\phi + 1) \\ &= \phi^{n-2} + \phi^{n-3} \\ &< F_n + F_{n-1} \qquad \text{por hipótese de indução} \\ &= F_{n+1}. \end{aligned}$$

Sendo $b \geq F_{\ell+1} > \phi^{\ell-1}$, segue-se que $\log b \geq \log(\phi^{\ell-1})$. Como $\log(\phi^{\ell-1}) = (\ell - 1)\log\phi$, temos que

$$\ell \leq \frac{\log b + \log\phi}{\log\phi} = \frac{\log b}{\log\phi} + 1.$$

observações.

1. O número de dígitos de b é dado por $\lfloor logb \rfloor + 1$.
2. O número de iterações do algoritmo de Euclides é majorado logaritmicamente pelo mais pequeno dos números a, b (o número de dígitos de a não vai influenciar o número de passos).
3. Tendo em conta que $\frac{1}{\log \phi} < 5$, podemos concluir que $\ell - 1 < 5 \log b$, o que nos permite provar o Teorema de Lamé (O número de divisões necessárias para encontrar o máximo divisor comum de dois inteiros positivos usando o algoritmo de Euclides não excede cinco vezes o número de dígitos decimais do menor dos dois inteiros.)

observação. O algoritmo de Euclides pode ser utilizado para exprimir o máximo divisor comum de dois inteiros como uma combinação linear desses inteiros (identidade de Bézout).

exemplo. Ilustramos este processo ao exprimir $\mathrm{m.d.c.}(252, 198) = 18$ como uma combinação linear de 252 e 198.

Referindo-nos aos passos do algoritmo de Euclides utilizados para determinar $\mathrm{m.d.c.}(252, 198)$, no penúltimo passo observamos que

$$18 = 54 - 1 \times 36.$$

Pelo passo anterior, temos que

$$36 = 198 - 3 \times 54,$$

o que implica que

$$18 = 54 - 1 \times (198 - 3 \times 54) = 4 \times 54 - 1 \times 198.$$

De modo semelhante, pelo primeiro passo, obtemos

$$54 = 252 - 1 \times 198,$$

pelo que

$$18 = 4(252 - 1 \times 198) - 1 \times 198 = 4 \times 252 - 5 \times 198.$$

Esta última equação apresenta $18 = \mathrm{m.d.c.}(252, 198)$ como uma combinação linear de 252 e 198.

em geral. Para ver como $\mathrm{m.d.c.}(a, b)$ pode ser expresso como combinação linear de a e b, consideremos a sucessão de equações gerada pelo algoritmo de Euclides.

Pela penúltima equação , obtemos

$$r_\ell = \mathrm{m.d.c.}(a, b) = r_{\ell-2} - r_{\ell-1}q_{\ell-1}.$$

Isto expressa $\mathrm{m.d.c.}(a, b)$ como uma combinação linear de $r_{\ell-2}$ e $r_{\ell-1}$. A equação imediatamente anterior pode ser usada para exprimir $r_{\ell-1}$ como

$$r_{\ell-1} = r_{\ell-3} - r_{\ell-2}q_{\ell-2}.$$

Substituímos esta expressão na equação anterior para $\mathrm{m.d.c.}(a, b)$, obtendo

$$\begin{aligned}\mathrm{m.d.c.}(a, b) &= r_{\ell-2} - (r_{\ell-3} - r_{\ell-2}q_{\ell-2})q_{\ell-1} \\ &= (1 + q_{\ell-1}q_{\ell-2})r_{\ell-2} - q_{\ell-1}r_{\ell-3}.\end{aligned}$$

Isto exprime $\mathrm{m.d.c.}(a, b)$ como uma combinação linear de $r_{\ell-2}$ e $r_{\ell-3}$.

Prosseguimos este processo, refazendo os cálculos na ordem inversa dos passos do algoritmo de Euclides, expressando $\mathrm{m.d.c.}(a, b)$ como combinação linear de cada par sucessivo de restos, até obtermos $\mathrm{m.d.c.}(a, b)$ como combinação linear de $r_0 = a$ e $r_1 = b$.

Especificamente, se numa determinada etapa tivermos

$$\mathrm{m.d.c.}(a,b) = sr_i + tr_{i-1},$$

e sabendo que

$$r_i = r_{i-2} - r_{i-1}q_{i-1},$$

segue-se que

$$\begin{aligned}\mathrm{m.d.c.}(a,b) &= s(r_{i-2} - r_{i-1}q_{i-1}) + tr_{i-1}\\ &= (t - sq_{i-1})r_{i-1} + sr_{i-2}.\end{aligned}$$

Este processo permite percorrer as equações geradas pelo algoritmo de Euclides de trás para a frente, garantindo que o máximo divisor comum de a e b pode ser expresso como combinação linear de a e b.

obervação. Este método para expressar $\mathrm{m.d.c.}(a,b)$ como combinação linear de a e b não é particularmente eficiente do ponto de vista computacional, especialmente para números grandes, pois exige a execução do algoritmo de Euclides, o registo de todos os seus passos e, posteriormente, a sua reversão para exprimir $\mathrm{m.d.c.}(a,b)$ como combinação linear de cada par sucessivo de restos, implicando armazenamento adicional e operações extra.

No entanto, existe um método alternativo para determinar $\mathrm{m.d.c.}(a, b)$ e expressá-lo como combinação linear de a e b que requer apenas uma única execução do algoritmo de Euclides.

O teorema seguinte apresenta este método, conhecido como algoritmo de Euclides estendido.

teorema. (algoritmo de Euclides estendido) Sejam a e b dois inteiros positivos tais que $a \geq b$. Então, o máximo divisor comum de a e b pode ser expresso como

$$\mathrm{m.d.c.}(a, b) = s_\ell a + t_\ell b,$$

onde s_ℓ e t_ℓ são os termos de ordem ℓ das sequências definidas recursivamente por

$$s_0 = 1, \quad t_0 = 0,$$

$$s_1 = 0, \quad t_1 = 1$$

e

$$s_i = s_{i-2} - q_{i-1} s_{i-1}, \quad t_i = t_{i-2} - q_{i-1} t_{i-1}$$

para $i = 2, 3, \ldots, \ell$, sendo q_i os quocientes obtidos nas divisões sucessivas do algoritmo de Euclides quando aplicado para determinar $\mathrm{m.d.c.}(a, b)$.

Antes de apresentar a prova do teorema anterior, vejamos um exemplo.

exemplo. Na tabela seguinte, resumimos os passos utilizados pelo algoritmo de Euclides estendido para expressar $\mathrm{m.d.c.}(252, 198)$ como combinação linear de 252 e 198:

i	r_i	r_{i+1}	q_{i+1}	r_{i+2}	s_i	t_i
0	252	198	1	54	1	0
1	198	54	3	36	0	1
2	54	36	1	18	1	-1
3	36	18	2	0	-3	4
4	18	0	-	-	4	-5

Os valores de s_i e t_i, para $i = 0, 1, 2, 3, 4$, são calculados da seguinte forma:

$$\begin{aligned}
&s_0 = 1, \quad s_1 = 0, && t_0 = 0, \quad t_1 = 1,\\
&s_2 = s_0 - s_1 q_1 = 1 - 0 \times 1 = 1, && t_2 = t_0 - t_1 q_1 = 0 - 1 \times 1 = -1,\\
&s_3 = s_1 - s_2 q_2 = 0 - 1 \times 3 = -3, && t_3 = t_1 - t_2 q_2 = 1 - (-1) \times 3 = 4,\\
&s_4 = s_2 - s_3 q_3 = 1 - (-3) \times 1 = 4, && t_4 = t_2 - t_3 q_3 = -1 - 4 \times 1 = -5.
\end{aligned}$$

Como $r_4 = 18 = \mathrm{m.d.c.}(252, 198)$ e $r_4 = s_4 a + t_4 b$, obtemos

$$18 = \mathrm{m.d.c.}(252, 198) = 4 \times 252 - 5 \times 198.$$

Apresentamos, de seguida, a demonstração do último teorema.

demonstração. (algoritmo de Euclides estendido) Mostremos que

$$r_i = s_i a + t_i b,$$

para todo $i = 0, 1, \ldots, \ell$. Como $\mathrm{m.d.c.}(a, b) = r_\ell$, uma vez estabelecida esta igualdade, poderemos concluir que

$$\mathrm{m.d.c.}(a, b) = s_\ell a + t_\ell b.$$

A demonstração desta igualdade segue por indução.

Para $i = 0$, temos:

$$a = r_0 = 1 \times a + 0 \times b = s_0 a + t_0 b.$$

Logo, a igualdade é válida para $i = 0$.

De modo semelhante, para $i = 1$, temos:

$$b = r_1 = 0 \times a + 1 \times b = s_1 a + t_1 b.$$

Assim, a igualdade verifica-se também para $i = 1$.

Seja $k \geq 2$ tal que

$$r_i = s_i a + t_i b,$$

para todo $i = 0, 1, \ldots, k-1$. Pelo passo k do algoritmo de Euclides, sabemos que

$$r_k = r_{k-2} - r_{k-1} q_{k-1}.$$

Pela hipótese de indução, obtemos:

$$r_k = (s_{k-2} a + t_{k-2} b) - (s_{k-1} a + t_{k-1} b) q_{k-1}.$$

Reagrupando os termos:

$$r_k = (s_{k-2} - s_{k-1} q_{k-1}) a + (t_{k-2} - t_{k-1} q_{k-1}) b.$$

Atendendo à definição das sequências $(s_i)_i$ e $(t_i)_i$, concluímos que

$$r_k = s_k a + t_k b,$$

o que completa a demonstração.

```
def myxgcd(r0, r1):
    s0, s1 = 1, 0
    t0, t1 = 0, 1
    print(s0,",t0)
    while r1 !=0 :
        q1 = r0//r1
        r_aux = r1
        r1 = r0%r1
        r0 = r_aux
        t2 = t0 - q1*t1
        t0 = t1
        t1 = t2
        s2 = s0 - q1*s1
        s0 = s1
        s1 = s2
        print(s0,",t0)
    return r0, s0, t0
```

Recordemos que, dados $a, b, c \in \mathbb{Z}$, com a e b não simultaneamente nulos, a equação diofantina

$$ax + by = c$$

tem solução se e só se $\mathrm{m.d.c.}(a, b) \mid c$.

Além disso, se $x = x_0$, $y = y_0$ é uma solução particular da equação, então todas as soluções são dadas por

$$x = x_0 + \frac{b}{d}t, \qquad y = y_0 - \frac{a}{d}t,$$

onde t é um inteiro e $d = \mathrm{m.d.c.}(a, b)$.

Admitamos que $d \mid c$. Então, existe $k \in \mathbb{Z}$ tal que $c = dk$.

Usando o algoritmo de Euclides, podemos obter inteiros x' e y' tais que $ax' + by' = d$.

Segue-se que $(ax' + by')k = dk$, ou seja,

$$a(x'k) + b(y'k) = c.$$

Logo, $x_0 = x'k$ e $y_0 = y'k$ é uma solução particular da equação diofantina considerada.

observação. Podemos usar o algoritmo de Euclides estendido para obter a solução particular, havendo ganho na eficiência/utilização da memória.

exemplo. Consideremos a equação diofantina $10021x + 342y = 15$.

Na tabela seguinte, resumimos os passos utilizados pelo algoritmo de Euclides estendido para expressar $\mathrm{m.d.c.}(10021, 342)$ como combinação linear de 10021 e 342:

i	r_i	r_{i+1}	q_{i+1}	r_{i+2}	s_i	t_i
0	10021	342	29	103	1	0
1	342	103	3	33	0	1
2	103	33	3	4	1	-29
3	33	4	8	1	-3	88
4	4	1	4	0	10	-293
5	1	0	-	-	-83	2432

Como $r_5 = 1 = \mathrm{m.d.c.}(10021, 342)$ e $r_5 = s_5 a + t_5 b$, obtemos

$$1 = \mathrm{m.d.c.}(10021, 342) = -83 \times 10021 + 2432 \times 342.$$

Assim, $x_0 = -83 \times 15 = -1245$, $y_0 = 2432 \times 15 = 36480$ é uma solução particular da equação diofantina $10021x + 342y = 15$.