

Hledání netriviálního dělitele

Předpokládejme, že máme dáno přirozené číslo N , o němž víme, že je složené. Naším úkolem je nalézt netriviálního dělitele čísla N .

Odhadněme nejprve časovou náročnost metody pokusného dělení: je třeba číslo N postupně vydělit všemi prvočísly nepřevyšujícími \sqrt{N} . Každé takové dělení zabere čas řádu $O(\ln^2 N)$, celá metoda je tedy řádu $O(N^{\frac{1}{2}} \ln^2 N)$.

První metoda, jejíž čas je lepší než právě uvedený, byla navržena Lehmannem. Je založena na následující větě.

Lehmannova metoda

Věta (Lehmann). Necht' N je liché přirozené číslo, $N = pq$, kde $\sqrt[3]{N} < p \leq q$ jsou prvočísla. Pak existují $x, y, k \in \mathbb{Z}$ splňující

- (1) $x^2 - y^2 = 4kN$, $1 \leq k \leq \lceil \sqrt[3]{N} \rceil$;
- (2) $2|k \Rightarrow x \equiv 1 \pmod{2}$; $2 \nmid k \Rightarrow x \equiv k + N \pmod{4}$;
- (3) $0 \leq x^2 - 4kN < \frac{N}{\lceil \sqrt[3]{N} \rceil}$, $x > 0$.

Jestliže naopak pro dané liché přirozené číslo $N = pq$, kde p, q jsou prvočísla, máme celá čísla x, y, k splňující podmínky (1), (2) a (3), pak jeden z největších společných dělitelů $(x + y, N)$ a $(x - y, N)$ je roven p a druhý q .

Rovněž platí, že je-li N liché prvočíslo, pak žádná trojice celých čísel x, y, k splňujících podmínky (1), (2) a (3) neexistuje.

Důkaz. Později si ukážeme důkaz pomocí teorie dobrých aproximací, který objevil Don Zagier.

Použití věty

Mějme dáno liché přirozené číslo N , o kterém je známo, že to není prvočíslo. Metodou pokusného dělení ověříme, že N není dělitelné prvočísly nepřevyšujícími $\sqrt[3]{N}$, anebo najdeme netriviálního dělitele. Tato část algoritmu je tedy řádu $O(N^{\frac{1}{3}} \ln^2 N)$. Pokud N nemá prvočíselného dělitele menšího než $\sqrt[3]{N}$, musí být tvaru $N = pq$, kde p, q jsou prvočísla.

Budeme pak postupně volit $k \in \{1, 2, \dots, \lceil \sqrt[3]{N} \rceil\}$ a pro každé takové k necháme x proběhnout všechna celá čísla splňující podmínky (2) a (3) z předchozí věty. Pro každé takové x pak testujeme, zda $x^2 - 4kN$ je druhá mocnina přirozeného čísla. Pokud ano, označíme $y = \sqrt{x^2 - 4kN}$ a spočítáme $(x + y, N)$, což je p nebo q . Je jasné, že časová náročnost algoritmu závisí na tom, jak rychle jsme schopni rozhodnout, zda přirozené číslo je nebo není druhou mocninou. Cesta vedoucí přes výpočet reálné odmocniny, zaokrouhlení a zkoušku jistě není ta pravá.

Algoritmus (Celočíslná druhá odmocnina). Pro dané přirozené číslo n algoritmus najde přirozené číslo m splňující $m^2 \leq n < (m + 1)^2$.

1. [Inicializace] Polož $x \leftarrow n$ (viz též diskusi za algoritmem).
2. [Krok] Pomocí celočíselného dělení a posunu spočítej $y \leftarrow [(x + \lfloor \frac{n}{x} \rfloor)/2]$.
3. [Konec?] Je-li $y < x$, polož $x \leftarrow y$ a jdi na 2. Jinak vytiskni x a skonči.

Důkaz algoritmu. Podle kroku 3 hodnota proměnné x klesá, algoritmus se tedy zastaví. Ukažme, že výsledek, který dává, je správný. Protože $x \in \mathbb{Z}$, je $[(x + \lfloor \frac{n}{x} \rfloor)/2] \leq (x + \lfloor \frac{n}{x} \rfloor)/2 \leq (x + \frac{n}{x})/2 < (x + \lfloor \frac{n}{x} \rfloor + 1)/2 \leq [(x + \lfloor \frac{n}{x} \rfloor)/2] + 1$, a tedy platí $[(x + \lfloor \frac{n}{x} \rfloor)/2] = [(x + \frac{n}{x})/2]$. Označme $q = \lfloor \sqrt{n} \rfloor$. Protože $\frac{1}{t}(t - \sqrt{n})^2 \geq 0$ pro libovolné $t > 0$, platí $\frac{1}{2}(t + \frac{n}{t}) \geq \sqrt{n}$, tedy $x \geq q$ je splněno v průběhu celého algoritmu. Předpokládejme, že se algoritmus zastavil, tj. že $y = [(x + \frac{n}{x})/2] \geq x$ a dokažme $x = q$. Předpokládejme $x \geq q + 1$. Pak $x > \sqrt{n}$ a platí

$$y - x = \left[\frac{1}{2} \left(x + \frac{n}{x} \right) \right] - x = \left[\frac{1}{2} \left(\frac{n}{x} - x \right) \right] = \left[\frac{1}{2x} (n - x^2) \right] < 0, \quad \text{spor.}$$

Časová náročnost celočíselné odmocniny

V kroku 1 je jistě výhodnější místo n zvolit číslo bližší \sqrt{n} . Vhodné může být např. zjistit řád e nejvyšší dvojkové cifry n , tj. přirozené číslo e splňující $2^e \leq n < 2^{e+1}$ a položit $x \leftarrow 2^{1+\lceil \frac{e}{2} \rceil}$. Pak totiž $x^2 \leq 2^{e+2} \leq 4n$, $x^2 \geq 2^{e+1} > n$, tj. $\sqrt{n} < x \leq 2\sqrt{n}$. Po provedení kroku 2 pak platí

$$\begin{aligned}x - y &= -\left[\frac{1}{2x}(n - x^2)\right] \geq -\frac{1}{2x}(n - x^2) = \\ &= \frac{1}{2x}(x + \sqrt{n})(x - \sqrt{n}) \geq \frac{1}{2x}\left(x + \frac{x}{2}\right)(x - \sqrt{n}) = \frac{3}{4}(x - \sqrt{n}).\end{aligned}$$

V každém dalším provedení kroku 3 se hodnota $x - \sqrt{n}$ zmenší alespoň čtyřikrát, neboť $y - \sqrt{n} = (x - \sqrt{n}) - (x - y) \leq \frac{1}{4}(x - \sqrt{n})$ a tedy krok 3 provádíme řádově $O(\ln n)$ -krát. Protože celočíselné dělení je řádu $O(\ln^2 n)$, je celý algoritmus řádu $O(\ln^3 n)$.

Zkrácení času výpočtu

Pokud nás, podobně jako v případě Lehmannova algoritmu, zajímá jen to, zda n je či není druhou mocninou přirozeného čísla, je možné rozhodování zrychlit: zjistíme, zda je n kvadratickým zbytkem modulo nějaké zvolené číslo m (tj. zda má řešení kongruence $x^2 \equiv n \pmod{m}$ – pokud n je druhou mocninou přirozeného čísla, tato kongruence řešení mít musí). Budeme postupovat takto: vydělíme číslo n číslem m se zbytkem a získaný zbytek porovnáme s tabulkou všech kvadratických zbytků modulo m , kterou budeme mít předem spočítanu v paměti. Vhodným modulem může být například číslo $1989 = 3^2 \cdot 13 \cdot 17$ nebo $1925 = 5^2 \cdot 7 \cdot 11$. Pravděpodobnost, že náhodně zvolené přirozené číslo je kvadratický zbytek modulo 1925, je $\frac{11}{25} \cdot \frac{4}{7} \cdot \frac{6}{11} = \frac{24}{175}$, pro modul 1989 dokonce je $\frac{4}{9} \cdot \frac{7}{13} \cdot \frac{9}{17} = \frac{28}{221}$. Provedeme-li test pro oba moduly, poběží předchozí algoritmus jen s pravděpodobností $\frac{96}{5525}$, tedy jen asi v 1,7% případů.

Toto vylepšení nebude mít vliv na asymptotickou časovou náročnost, sníží však významně O -konstantu.

Algoritmus (Naplnění tabulek kvadratických zbytků).

Algoritmus sestaví vektory T_1 o délce 1989 a T_2 o délce 1925 tak, že pro každé $0 \leq i \leq 1988$ platí $T_1[i] = 1$, právě když kongruence $x^2 \equiv i \pmod{1989}$ má řešení, a pro každé $0 \leq i \leq 1924$ platí $T_2[i] = 1$, právě když kongruence $x^2 \equiv i \pmod{1925}$ má řešení.

- 1. [Naplnění T_1] Pro i od 0 po 1988 polož $T_1[i] \leftarrow 0$. Pak pro i od 0 po 994 polož $T_1[i^2 \bmod 1989] \leftarrow 1$.*
- 2. [Naplnění T_2] Pro i od 0 po 1924 polož $T_2[i] \leftarrow 0$. Pak pro i od 0 po 962 polož $T_2[i^2 \bmod 1925] \leftarrow 1$.*

Algoritmus (Test na čtverec). Pro dané přirozené číslo n algoritmus zjistí, zda je n druhá mocnina přirozeného čísla, a pokud ano, vytiskne \sqrt{n} .

1. [Test na 1989] Polož $r \leftarrow n \bmod 1989$. Je-li $T_1[r] = 0$, odpověz, že n není druhá mocnina přirozeného čísla a skonči.
2. [Test na 1925] Polož $r \leftarrow n \bmod 1925$. Je-li $T_2[r] = 0$, odpověz, že n není druhá mocnina přirozeného čísla a skonči.
3. [Spočítej odmocninu] Algoritmem celočíselné druhé odmocniny spočítej $m = \lfloor \sqrt{n} \rfloor$. Je-li $n \neq m^2$, odpověz, že n není druhá mocnina přirozeného čísla a skonči. Jinak odpověz, že n je druhá mocnina přirozeného čísla m a skonči.

Časová náročnost Lehmannova algoritmu

Odhadněme počet hodnot, které musíme za x dosazovat pro pevně zvolené $k \in \{1, 2, \dots, \lceil \sqrt[3]{N} \rceil\}$. Odhadneme-li $x + 2\sqrt{kN} \geq 4\sqrt{kN}$ pomocí (3) ve výrazu

$$x - 2\sqrt{kN} = \frac{x^2 - 4kN}{x + 2\sqrt{kN}} < \frac{N}{\lceil \sqrt[3]{N} \rceil} \cdot \frac{1}{4\sqrt{kN}} = \frac{1}{4\lceil \sqrt[3]{N} \rceil} \cdot \sqrt{\frac{N}{k}},$$

dostáváme, že x splňuje

$$2\sqrt{kN} \leq x < 2\sqrt{kN} + \frac{1}{4\lceil \sqrt[3]{N} \rceil} \cdot \sqrt{\frac{N}{k}},$$

patří tedy x do intervalu délky $\frac{1}{4\lceil \sqrt[3]{N} \rceil} \sqrt{\frac{N}{k}}$. Délka intervalu je řádu $O(k^{-\frac{1}{2}} N^{\frac{1}{6}})$, pro pevné k je tedy časová náročnost algoritmu řádu $O(k^{-\frac{1}{2}} N^{\frac{1}{6}} \ln^3 N)$.

Sečtením přes všechna k dostáváme, že celková časová náročnost je řádu

$$O\left(N^{\frac{1}{6}} \ln^3 N \sum_{k=1}^{\lceil \sqrt[3]{N} \rceil} k^{-\frac{1}{2}}\right).$$

Přitom $\int_1^r k^{-\frac{1}{2}} dk = [2k^{\frac{1}{2}}]_1^r = 2\sqrt{r} - 2$, volbou $r = \sqrt[3]{N}$ upravíme řád časové náročnosti hledání čísel k , x , y do tvaru

$$O\left(N^{\frac{1}{6}} \ln^3 N \cdot \sqrt{N^{\frac{1}{3}}}\right) = O(N^{\frac{1}{3}} \ln^3 N).$$

Protože časová náročnost první části algoritmu, totiž metody pokusného dělení čísla nepřevyšujícími $\sqrt[3]{N}$, je řádu $O(N^{\frac{1}{3}} \ln^2 N)$, je celková časová náročnost Lehmannova algoritmu řádu $O(N^{\frac{1}{3}} \ln^3 N)$. Lehmannova metoda je tedy asymptoticky výrazně lepší než algoritmus pokusného dělení, jehož časová náročnost je řádu $O(N^{\frac{1}{2}} \ln^2 N)$.

Další metoda hledání netriviálního dělitele:

Pollardova ρ metoda

Předpokládejme, že M je konečná množina a $f : M \rightarrow M$ zobrazení. Zvolme $x_0 \in M$ a pro každé $n \in \mathbb{N}$ položme $x_n = f(x_{n-1})$. Protože je M konečná, v posloupnosti $(x_n)_{n=0}^{\infty}$ nemohou být všechny prvky různé.

Nechť $i \in \mathbb{N} \cup \{0\}$ je nejmenší index, pro který existuje nějaký index $n > i$ s vlastností $x_i = x_n$. Dále označme j nejmenší takové n . Pak i nazýváme předperioda a $j - i$ perioda posloupnosti $(x_n)_{n=0}^{\infty}$.

Je možné dokázat, že střední hodnota předperiody i a periody (mají-li všechny dvojice $(x_0, f) \in M \times M^M$ stejnou pravděpodobnost) je řádu $O(\sqrt{|M|})$.

Základní myšlenka Pollardovy ρ metody

Nechť $f(x)$ je mnohočlen s celými koeficienty. Hledáme (neznámého) prvočíselného dělitele přirozeného čísla N , o kterém víme, že je složené. Zvolme celé číslo x_0 a počítejme $x_n = f(x_{n-1}) \bmod N$. Pak ovšem $y_n = x_n \bmod p$ vyhovuje téže rekurzi modulo p . Pokud se f chová jako náhodné zobrazení (což nevíme, ale budeme to předpokládat), je předperioda a perioda posloupnosti $(y_n)_{n=0}^{\infty}$ řádu $O(\sqrt{p})$, kdežto předperioda a perioda posloupnosti $(x_n)_{n=0}^{\infty}$ je řádu $O(\sqrt{N})$. Dá se tedy čekat, že existují $i < j$ tak, že $y_i = y_j$, ale $x_i \neq x_j$. Pak ovšem je $(x_i - x_j, N)$ netriviální dělitel čísla N .

Je nutné nějak zvolit x_0 a f . Volba x_0 se zdá být nepodstatná, ne však volba f . Je vhodné, aby f byl jednoduchý polynom pro výpočet, konstantní ani lineární však nejsou vhodné. Budeme tedy volit f jako co nejjednodušší kvadratický polynom. Je ověřeno experimentálně, že polynomy $f = x^2$ a $f = x^2 - 2$ nejsou vhodné, kdežto $f = x^2 + c$, kde $c \neq 0$ a $c \neq -2$, pracuje docela dobře, i když nejsme schopni určit ani periodu ani předperiodu.

Implementace Pollardovy ρ metody

Je jasné, že uchovávání všech již vypočtených členů posloupnosti $(x_n)_{n=0}^{\infty}$ a jejich neustálé porovnávání s nově vypočtenou hodnotou by bylo velmi zdoluhavé. Jednoduchou metodou, jak se tomuto zdoluhavému výpočtu vyhnout, je porovnávat postupně x_n a x_{2n} . Pak totiž prvočíselného dělitele p čísla N objevíme nejpozději po k krocích, kde k je součet předperiody a periody posloupnosti modulo p . Znamená to počítat iterace dvou posloupností: položit $z_0 = x_0$, iterovat $x_n = f(x_{n-1}) \bmod N$ a $z_n = f(f(z_{n-1})) \bmod N$ a počítat $(x_n - z_n, N)$.

Za (nedokázaného) předpokladu, že f se chová jako náhodné zobrazení, je počet nutných kroků $O(\sqrt{p})$. V každém kroku počítáme třikrát f , dvakrát zbytek po dělení N a jednou největší společný dělitel, vše je $O(\ln^2 N)$. Celková časová náročnost je tedy $O(\sqrt{p} \ln^2 N)$, což vzhledem k $p \leq \sqrt{N}$ dává $O(\sqrt[4]{N} \ln^2 N)$. Je vhodné si uvědomit, že podobně jako metoda postupného dělení je i tato metoda citlivá k velikosti prvočíselných dělitelů – „malé“ dělitele čísla N odstraňuje rychleji než „velké“.

Další metoda hledání netriviálního dělitele:

Pollardova $p - 1$ metoda

Tato metoda je schopna najít i značně velké prvočíselné dělitele p čísla N , pokud $p - 1$ není dělitelné příliš velkou mocninou prvočísla.

Definice. Necht' B je přirozené číslo. Řekneme, že přirozené číslo n je B -hladké, jestliže pro libovolné prvočísla p a libovolné přirozené číslo k platí

$$p^k \mid n \quad \Rightarrow \quad p^k \leq B.$$

Příklad. Víme, že pro každé $n \in \mathbb{N}$ platí, že $\binom{2n}{n}$ je $2n$ -hladké. Dokázali jsme totiž už dříve, že platí

Lemma 2. Pro libovolné přirozené číslo n a libovolné prvočísla p platí: je-li $\ell = \nu_p\left(\binom{2n}{n}\right)$, pak $p^\ell \leq 2n$.

Základní myšlenka Pollardovy $p - 1$ metody

Přepokládejme, že pro nějaký prvočíselný dělitel p čísla N platí, že číslo $p - 1$ je B -hladké pro nějaké nepříliš velké přirozené číslo B . Zvolme libovolně $1 < a < N$. Je-li $(a, N) > 1$, jsme hotovi. Budeme proto předpokládat, že $(a, N) = 1$. Pak podle definice číslo $p - 1$ dělí nejmenší společný násobek L_B čísel $1, 2, 3, \dots, B$. Z Fermatovy věty pak plyne $a^{L_B} \equiv 1 \pmod{p}$ a tedy $(a^{L_B} - 1, N) > 1$. Budeme tedy testovat poslední podmínku pro zvyšující se hodnoty exponentu $e \mid L_B$ (budeme postupně umocňovat na faktory z kanonického rozkladu čísla L_B). Je velmi nepravděpodobné, že poprvé, kdy platí $(a^e - 1, N) > 1$, je tento největší společný dělitel roven N . Může se ovšem stejně stát, že metoda selže, jestliže pro žádné prvočíselno $p \mid N$ číslo $p - 1$ není B -hladké.

Při výpočtu zabere nejvíce času výpočet největšího společného dělitele, proto budeme postupovat tak, že budeme uchovávat součiny a počítat největší společný dělitel jen čas od času.

Algoritmus (Pollardova $p - 1$ metoda, první stádium). Dáno složené N a hranice B , hledáme netriviálního dělitele N . Máme tabulku $p[1], p[2], \dots, p[k]$ všech prvočísel menších nebo rovných B .

1. [Inicializace] Polož $x \leftarrow 2, y \leftarrow x, P \leftarrow 1, c \leftarrow 0, i \leftarrow 0, j \leftarrow i$.
2. [Další prvočíslo] Polož $i \leftarrow i + 1$. Je-li $i > k$, spočti největší společný dělitel $g \leftarrow (P, N)$. Je-li $g = 1$, napiš, že jsi neuspěl a skonči, jinak polož $i \leftarrow j, x \leftarrow y$ a jdi na 5. Jinak (tj. pro $i \leq k$) polož $q \leftarrow p[i], q_1 \leftarrow q, \ell \leftarrow \lfloor \frac{B}{q} \rfloor$.
3. [Spočti mocninu] Dokud $q_1 \leq \ell$, dělej $q_1 \leftarrow q_1 \cdot q$. Pak polož $x \leftarrow x^{q_1} \bmod N, P \leftarrow P \cdot (x - 1) \bmod N, c \leftarrow c + 1$ a je-li $c < 20$, jdi na 2.
4. [Největší společný dělitel] Polož $g \leftarrow (P, N)$. Je-li $g = 1$, polož $c \leftarrow 0, j \leftarrow i, y \leftarrow x$ a jdi na 2. Jinak polož $i \leftarrow j, x \leftarrow y$.
5. [Počítej znovu] Polož $i \leftarrow i + 1, q \leftarrow p[i], q_1 \leftarrow q$.
6. [Skončil jsi?] Polož $x \leftarrow x^q \bmod N, g \leftarrow (x - 1, N)$. Je-li $g = 1$, polož $q_1 \leftarrow q \cdot q_1$ a je-li $q_1 < B$, jdi na 6, jinak jdi na 5. Jinak (tj. pro $g > 1$), je-li $g < N$, vytiskni g a skonči. Konečně, je-li $g = N$ (velmi nepravděpodobné), napiš, že jsi neuspěl a skonči.

Pokud algoritmus selhal v bodě 6, znamená to, že všechna prvočísla p dělicí N byla nalezena současně, což je značně nepravděpodobné. Může proto mít smysl zkusit tentýž algoritmus s jinou počáteční hodnotou (např. $x \leftarrow 3$).

I v této jednoduché formě jsou výsledky algoritmu působivé. Samozřejmě, jsou-li $p < q$ prvočísla zhruba stejně velká taková, že i $2p + 1$ a $2q + 1$ jsou prvočísla, pro $N = (2p + 1)(2q + 1)$ by algoritmus rozložil N jen pro $B \geq p$. Uspěl by tedy za dobu srovnatelnou s algoritmem pokusného dělení.

Obvyklé hodnoty B jsou mezi 10^5 a 10^6 .

Druhé stádium Pollardovy $p - 1$ metody

Požadavek, aby existovalo prvočíslo $p \mid N$ takové, že $p - 1$ je B -hladké, je poměrně silný. Má proto smysl jej zeslabit a požadovat jen, aby bylo $p - 1$ zcela rozloženo po pokusném dělení do hranice B , tj. požadovat, aby $p - 1 = f \cdot q$, kde f je B -hladké a q je prvočíslo větší než B (ale zase ne příliš velké). Pro naše účely budeme předpokládat, že f je B_1 -hladké a prvočíslo q splňuje $B_1 < q \leq B_2$, kde B_1 je naše staré B a B_2 je o dost větší konstanta. Samozřejmě, že bychom p objevili i předchozím algoritmem pro $B = B_2$, ale to by trvalo příliš dlouho.

Podobně jako předtím nyní platí $(a^{qL_B} - 1, N) > 1$. Budeme postupovat takto: po ukončení prvního stadia (tj. předchozího algoritmu) máme spočítáno $b = a^{L_B} \bmod N$. Předpokládejme, že máme uloženy rozdíly prvočísel od B_1 do B_2 . Tyto rozdíly jsou malé a je jich nemnoho. Můžeme proto snadno předpočítat b^d pro všechny možné rozdíly d a získat b^q postupným donásobováním původní mocniny b předpočítanými hodnotami b^d . Znamená to, že pro každé prvočíslo mezi B_1 a B_2 nahradíme umocňování pouhým násobením, které je samozřejmě mnohem rychlejší.

Algoritmus (Pollardova $p - 1$ metoda, druhé stadium). Dáno složené N a hranice B_1 a B_2 , hledáme netriviálního dělitele N . Máme tabulku $p[1], p[2], \dots, p[k_1]$ všech prvočísel menších nebo rovných B_1 a tabulku $d[1], d[2], \dots, d[k_2]$ všech diferencí prvočísel mezi B_1 a B_2 tak, že $d[1] = p[k_1 + 1] - p[k_1]$ atd.

1. [První stadium] Pro $B = B_1$ (a $k = k_1$) zkus rozložit N pomocí předchozího algoritmu. Jestliže tento algoritmus uspěje, skončí. Jinak tento algoritmus dal x . Polož $b \leftarrow x$, $P \leftarrow 1$, $c \leftarrow 0$, $i \leftarrow 0$, $j \leftarrow i$.
2. [Předpočítání] Pro všechny hodnoty rozdílů $d[i]$ (které jsou malé a je jich málo) spočítej a ulož $b^{d[i]}$. Polož $x \leftarrow x^{p[k_1]} \bmod N$, $y \leftarrow x$.
3. [Vpřed] Polož $i \leftarrow i + 1$, $x \leftarrow x \cdot b^{d[i]}$ (pomocí předpočítané hodnoty $b^{d[i]}$), $P \leftarrow P \cdot (x - 1) \bmod N$, $c \leftarrow c + 1$. Je-li $i \geq k_2$, jdi na 6. Jinak, je-li $c < 20$, jdi na 3.
4. [Největší společný dělitel] Polož $g \leftarrow (P, N)$. Je-li $g = 1$, polož $c \leftarrow 0$, $j \leftarrow i$, $y \leftarrow x$ a jdi na 3.
5. [Počítej znovu] Polož $i \leftarrow j$, $x \leftarrow y$. Pak opakuj $x \leftarrow x \cdot b^{d[i]}$, $i \leftarrow i + 1$, $g \leftarrow (x - 1, N)$ dokud nenastane $g > 1$ (což musí nastat). Je-li $g < N$, vytiskni g a skončí. Jinak (tj. je-li $g = N$, což je velmi nepravděpodobné), napiš, že jsi neuspěl a skončí.
6. [Neuspěl jsi?] Polož $g \leftarrow (P, N)$. Je-li $g > 1$, jdi na 5. V opačném případě (tj. je-li $g = 1$), napiš, že jsi neuspěl a skončí.

V případě nepravděpodobného neúspěšného konce v kroku 5 by také bylo možné začít znovu první stadium pro $x \leftarrow 3$ místo $x \leftarrow 2$ v kroku 1.

V této formě je algoritmus mnohem efektivnější než ve formě pouze prvního stadia. Obvyklé hodnoty konstant jsou $B_1 = 2 \cdot 10^6$, $B_2 = 10^8$.

Algoritmus je založen na aritmetice grupy \mathbb{F}_p^\times . Podobně lze pracovat i v $\mathbb{F}_{p^2}^\times/\mathbb{F}_p^\times$, v tomto případě je požadována B -hladkost čísla $p + 1$ místo $p - 1$.

Bylo by možné samozřejmě pracovat i v $\mathbb{F}_{p^4}^\times/\mathbb{F}_{p^2}^\times$ nebo $\mathbb{F}_{p^3}^\times/\mathbb{F}_p^\times$ nebo $\mathbb{F}_{p^6}^\times/(\mathbb{F}_{p^2}^\times \cdot \mathbb{F}_{p^3}^\times)$ s požadavkem B -hladkosti čísla $p^2 + 1$ nebo $p^2 + p + 1$ nebo $p^2 - p + 1$. To už jsou ale mnohem větší čísla a splnění požadavku B -hladkosti těchto čísel je méně pravděpodobné. Potřebujeme proto další grupy, jejichž řád je zhruba p , ve kterých jsme schopni pracovat (aniž známe prvočíslo p). Takovými grupami jsou grupy eliptických křivek nad \mathbb{F}_p .