Obsah

Úvod

Prvočísla sa v matematike vyskytovali už od staroveku. Majú kľúčový význam v teórii čísel, ktorá sa dodnes venuje otázkam súvisiacim napríklad s rozložením prvočísel, algoritmami na hľadanie prvočíselného rozkladu a na testovanie prvočíselnosti. Tieto testy našli svoje uplatnenie aj v praxi. V niektorých kryptosystémoch, ako napríklad RSA, sa na vytvorenie kľúča používajú veľké prvočísla. A práve pri hľadaní veľkých prvočísel sa môžu uplatniť rôzne algoritmy na testovanie prvočíselnosti.

V tejto práci sa budeme zaoberať AKS-algoritmom. AKS-algoritmus je prvý známy bezpodmienečne deterministický polynomiálny algoritmus na testovanie prvočíselnosti. V prvej kapitole uvedieme rôzne testy používané na testovanie prvočíselnosti a porovnáme ich s AKS-algoritmom. Už skôr boli známe niektoré algoritmy, ktoré mali polynomiálnu alebo „takmer“ polynomiálnu zložitosť. Išlo však buď o pravdepodobnostné algoritmy (t.j. algoritmus mohol niekedy odpovedať chybne, hoci s malou pravdepodobnosťou) alebo odvodenie polynomiálne časovej zložitosti záviselo napríklad na rozšírenej Riemannovej hypotéze (v prípade Rabin-Millerovho testu).

Až v roku 2002 objavili indickí matematici M. Agrawal, N. Kayal a N. Saxena deterministický algoritmus, ktorého časová zložitosť sa dá ohraničiť polynómom bez toho, že by sme museli prijať akúkoľvek dodatočnú hypotézu. Ich pôvodný dôkaz používal pomerne hlboké výsledky z analytickej teórie čísel, neskôr sa však podarilo zdôvodniť jeho správnosť a polynomiálnu zložitosť aj pomerne elementárnymi prostriedkami.

Pokúsime sa o teoretické odvodenie správnosti a zložitosti AKS-algoritmu, pričom sa budeme snažiť použiť čo najelementárnejší matematický aparát. Tento algoritmus budeme aj implementovať - konkrétne jednu z jeho viacerých verzií, ktoré sa vyskytli v literatúre.

Ako základnú literatúru popisujúcu AKS-algoritmus použijeme v prvom rade článok [2], z ktorého tento algoritmus pochádza, ale takisto aj článok [31] a poslednú kapitolu z knihy [30], ktorá sa tiež zaoberá týmto algoritmom.

Prvá kapitola stručne popisuje najznámejšie algoritmy používané na testovanie prvočíselnosti. V druhej kapitole tejto práce zhrnieme potrebné poznatky z algebry, teórie čísel a zložitosti algoritmov používaných v teórii čísel. Čitateľ, ktorý tieto poznatky ovláda, môže túto kapitolu vynechať a pokračovať priamo odvodením správnosti a časovej zložitosti AKS-algoritmu, ktoré je obsahom tretej a štvrtej kapitoly. Využijeme sme hlavne článok [31], v ktorom sú uvedené potrebné tvrdenia zo všetkých troch spomínaných oblastí. Niektoré pomocné poznatky z algebry možno nájsť aj v učebnici [21]. Výborným zdrojom pre vlastnosti prvočísel a niektoré odhady sú knihy [18], [23] ako aj kurz [19] a diplomová práca [32]. Teóriou čísel z algoritmického hľadiska sa zaoberajú napríklad knihy [14], [22] a [30]. Súčasťou práce je aj implementácia AKS-algoritmu. Niektorým detailom implementácie sme sa venovali v závere práce.

Kapitola 1

Prehľad algoritmov používaných na testovanie prvočíselnosti

Cieľom tejto kapitoly je stručne popísať staršie algoritmy na zisťovanie prvočíselnosti a zistiť aké výhody a nedostatky má v porovnaní s nimi AKS-algoritmus.

1.1 Testy prvočíselnosti

1.1.1 Lucas-Lehmerov test

Lucas-Lehmerov test patrí medzi tzv. $n+1$ testy - môže sa použiť, ak poznáme kanonický rozklad čísla $n+1$. Číslo $n+1$ má veľmi jednoduchý kanonický rozklad napríklad ak $n = 2^n - 1$ - takéto čísla nazývame *Mersennove čísla*. V prípade, že ide o prvočíslo, tak hovoríme o *Mersennových prvočíslach.* Dodnes otvoreným problémom v teórii čísel je otázka, či existuje nekonečne veľa Mersennových prvočísel.

Práve pomocou tohto testu sa podarilo nájsť väčšinu z dnes známych najväčších prvočísel. Tento test využíva tzv. Lucasove postupnosti, pre ktoré platia rekurzívne vzťahy umožňujúci ich rýchly výpočet.

Pre Mersennove prvočísla možno konkrétne použiť postupnosť danú vzťahmi $v_0 = 4$ a $v_{k+1} = v_k^2 - 2$. Platí, že $M_p = 2^p - 1$ je prvočíslo práve vtedy, keď $v_{p-2} \equiv 0 \pmod{M_p}$.

Lucas-Lehmerov test sa prvýkrát objavil v článku [24].

1.1.2 Rabin-Millerov test

Rabin-Millerov test je veľmi rýchly pravdepodobnostný algoritmus na zisťovanie prvočíselnosti. Ak tento algoritmus zistí, že dané číslo je zložené, číslo je skutočne zložené a algoritmus nám poskytne aj tzv. „svedka“ - t.j. číslo, pomocou ktorého môžeme ľahko overiť, že pôvodný vstup je skutočne zložené číslo. Ak algoritmus ako odpoveď vráti, že n je prvočíslo, nemusí to byť vždy pravda. Ale za cenu zvýšenia počtu iterácií algoritmu môžeme dosiahnuť ľubovoľne malú pravdepodobnosť, že nastane chyba.

Dá sa ukázať, že ak n je prvočíslo a $n-1 = 2^s d$, tak platí pre každé $a = 1, \ldots, n-1$

$$a^d \equiv 1 \,(\mathrm{mod}\ n) \qquad \text{alebo} \qquad a^{2^r t} \equiv -1 \,(\mathrm{mod}\ n) \text{ pre niektoré } r = 0, \ldots, s-1. \quad (1.1)$$

To znamená, že ak pre nejaké a zistíme, že niektorá z kongruencií neplatí, toto číslo a môžeme nazvať „svedkom“ - pomocou neho môžeme totiž ľahko overiť, že n je zložené.

Samozrejme overenie rovnosti (1.1) pre každé prirodzené číslo a menšie ako n by viedlo k exponenciálnemu algoritmu. Podarilo sa však ukázať, že pravdepodobnosť toho, že pre dané zložené číslo n a náhodne vybrané číslo a táto rovnosť platí je menej než $\frac{1}{4}$. Viacnásobným opakovaním testu môžeme teda znížiť pravdepodobnosť chyby na ľubovoľne malú - aj keď nikdy nemôžeme s úplnou istotou tvrdiť, že tento algoritmus vrátil prvočíslo. Autorom tohto pravdepodobnostného testu je Rabin [28], ktorý zmodifikoval Millerov výsledok z [25].

Millerovi sa podarilo ukázať, že ak platí rozšírená Riemannova hypotéza (ERH), tak stačí overovať rovnosť (1.1) pre $a < 2\log^2 n$ - za predpokladu ERH teda takto dostaneme polynomiálny deterministický algoritmus pre zisťovanie prvočíselnosti.

1.1.3 ECPP

Často používaným algoritmom je aj testovanie prvočíselnosti pomocou eliptických kriviek (Elliptic Curves Primality Proving). Používanie eliptických kriviek na testovanie prvočíselnosti zaviedli S. Goldwasser, J. Killian a A. O. L. Atkin ([3], [4], [17]).

Eliptická krivka rodu 1 je krivka daná rovnicou

$$y^2 = x^3 + ax + b,$$

pričom $4a^3 + 27b^2 \neq 0$. Na bodoch tejto krivky s racionálnymi súradnicami možno zaviesť operáciu sčitovania tak, že dostaneme grupu. Ak namiesto sčitovania použijeme sčitovanie modulo p, dostaneme grupu $E(a,b)/p$.

Táto grupa môže byť použitá podobným spôsobom ako grupa pozostávajúca z prvkov $\mathbb{Z}_n$, ktoré sú nesúdeliteľné s n, v testoch založených na malej Fermatovej vete. Malá Fermatova veta vlastne hovorí, že táto grupa má $n-1$ prvkov. V prípade ECPP algoritmu sa využije odhad na počet prvkov grupy $E(a,b)/p$

$$p + 1 - 2\sqrt{p} < |E(a,b)/p| < p + 1 + 2\sqrt{p}.$$

Tento test dosahuje očakávaný čas $O(\lg^{4+\varepsilon} n)$ a je polynomiálny na skoro všetkých vstupoch. ECPP dokazuje prvočíselnosť, čiže keď zastaví a odpovie, že vstup je prvočíslo, skutočne ide o prvočíslo.

1.1.4 APR a APRCL

Test, ktorý vytvorili v roku 1979 L. M. Adleman, C. Pomerance a R. S. Rumely [1] a neskôr bol zmodifikovaný na efektívnejšiu verziu H. Cohenom a A. K. Lenstrom [13] je vhodné spomenúť hlavne z toho dôvodu, že sa odhadom na časovú zložitosť najviac blíži k polynomiálnemu algoritmu. Tento test má zložitosť $O((\lg n)^{c \lg\lg\lg n})$. Dalo by sa povedať, že je to „skoro lineárna“ zložitosť, pretože exponent je extrémne pomaly rastúca funkcia.

1.2 Porovnanie s AKS-algoritmom

Ako už bolo spomenuté, AKS-algoritmus má polynomiálnu zložitosť pre ľubovoľný vstup. To sa nepodarilo ukázať o žiadnom z algoritmov vymenovaných v predošlej časti. Rabin-Millerov test má polynomiálnu zložitosť za predpokladu ERH, ECPP dosahuje polynomiálnu zložitosť pre „väčšinu“ vstupov (a predpokladá sa, že je polynomiálny aj pre ostatné vstupy) a APR, resp. APRCL-test je „takmer polynomiálny“. AKS-algoritmus je prvý test prvočíselnosti, o ktorom bolo bez akýchkoľvek dodatočných hypotéz dokázané, že je polynomiálny.

V porovnaní s doteraz spomenutými testami je veľkou výhodou AKS-algoritmu aj oveľa jednoduchší matematický aparát, ktorý je použitý. Správnosť a časovú zložitosť AKS-algoritmu je možné - ako neskôr uvidíme - odvodiť použitím pomerne elementárnych poznatkov z algebry a teórie čísel. Ide vlastne o isté zovšeobecnenie malej Fermatovej vety. Testy, ktoré sme uviedli v predchádzajúcej časti, využívajú oveľa zložitejší aparát. V prípade Lucas-Lehmerovho testu ide o niektoré vlastnosti Jacobiho symbolov. Dôkaz správnosti APR-testu a APRCL-testu sa opiera hlavne o zákony reciprocity a aritmetické vlastnosti tzv. Jacobiho súm. Obidva tieto algoritmy teda vyžadujú oveľa hlbšie poznatky z teórie čísel než sú potrebné na zvládnutie AKS-algoritmu. ECPP zasa používa výsledky komplexnej analýzy.

Zložitosť týchto algoritmov možno porovnať aj na základe veľmi jednoduchého kritéria - porovnaním dĺžok článkov, v ktorých sú uverejnené. Kým článok [2], v ktorom je odvodený AKS-algoritmus, má 9 strán, dĺžka článkov popisujúcich ostatné spomínané algoritmy sa pohybuje väčšinou okolo 30 strán.

Teraz stručne popíšeme jednotlivé kroky v AKS-algoritme (hoci tento popis AKS-algoritmu uvedieme ešte raz, keď budeme dokazovať jeho správnosť a časovú zložitosť).

1. Ak $n = a^b$ pre nejaké prirodzené čísla $a, b > 1$, tak vrátime `FALSE`.
2. Hľadáme najmenšie prvočíslo r také, že platí buď $(n, r) > 1$ alebo $(n, r) = 1$ a multiplikatívny rád d čísla n modulo r je viac než $4 \lg^2 n$.
3. Ak $r = n$, tak vrátime `TRUE`.
4. Ak $(n, r) > 1$ tak vrátime `FALSE`.
5. Postupne pre a od 1 po $\ell = \lfloor 2\sqrt{r} \lg n \rfloor + 1$ testujeme rovnosť
$$(x - a)^n = x^n - a \mod (x^r - 1)$$
v $\mathbb{Z}_n[x]$. Akonáhle nájdeme najmenšie a, pre ktoré táto rovnosť neplatí, vrátime `FALSE`.
6. Ak rovnosť platí pre všetky $a = 1, \ldots, \ell$, tak vrátime `TRUE`.

AKS-algoritmus má (aspoň v súčasnej podobe) skôr teoretický než praktický význam. Spomínané pravdepodobnostné algoritmy sú totiž oveľa efektívnejšie a pravdepodobnosť chyby je veľmi malá. Dajú sa použiť aj v situáciach, kde vadí malá pravdepodobnosť chyby. Existujú totiž algoritmy, ktoré vygenerujú tzv. „svedka“, pomocou ktorého je možné oveľa rýchlejšie overiť, že pôvodne zadaný vstup je prvočíslo.

Napriek tomu je snaha vylepšovať AKS-algoritmus z hľadiska časovej zložitosti, prípadne ho používať v kombinácii s inými metódami alebo použiť ideu AKS-algoritmu na vytvorenie pravdepodobných testov (pozri napríklad [10], [11], [20], [33]).

Kapitola 2

Pomocné tvrdenia

V tejto kapitole uvedieme niektoré pojmy a tvrdenia z algebry, teórie čísel a niektoré odhady zložitosti číselných algoritmov, ktoré budú neskôr potrebné pri teoretickom odvodení AKS-algoritmu a odhade jeho zložitosti.

2.1 Teória čísel

Označenie $\lg n$ budeme používať pre dvojkový logaritmus čísla n. Prirodzený logaritmus čísla n budeme značiť $\ln n$.

Budeme používať obvyklé označenie pre popis asymptotickej veľkosti funkcií. Budeme uvažovať funkcie, ktorých definičným oborom sú prirodzené čísla alebo kladné reálne čísla. Symbol $O(g(x))$ označuje množinu všetkých funkcií $f(x)$ takých, že existujú $M > 0$ a x_0, pre ktoré $x > x_0 \Rightarrow f(x) \leq Mg(x)$. Množina $o(g(x))$ obsahuje práve funkcie s vlastnosťou $\lim_{x\to\infty} \frac{f(x)}{g(x)} = 0$. Ako $\tilde{O}(g(x))$ budeme značiť množinu všetkých funkcií patriacich do $O(g(x)p(\lg g(x)))$, kde p je ľubovoľný polynóm. Namiesto označenia $f(x) \in O(g(x))$ používame zápis $f(x) = O(g(x))$, podobne pre $o(g(x))$, $\tilde{O}(g(x))$.

2.1.1 Malá Fermatova veta

Malá Fermatova veta (ktorá je špeciálnym prípadom Eulerovej vety) bola využitá vo viacerých algoritmoch na testovanie prvočíselnosti, hoci neposkytuje úplnú charakterizáciu prvočísel. Aj v AKS-algoritme sa využíva isté zovšeobecnenie tejto vety, preto uvedieme najprv jej dôkaz.

Lema 2.1.1. *Nech p je prvočíslo a nech $1 \leq i \leq p-1$. Potom*

$$\binom{p}{i} \equiv 0 \pmod{p}.$$

Dôkaz. Z kombinatorického významu čísla $\binom{p}{i}$ (počet i-prvkových podmnožín ľubovoľnej p-prvkovej množiny) vyplýva, že je to celé číslo. Máme vyjadrenie

$$\binom{p}{i} = \frac{p(p-1)\cdots(p-i+1)}{1\cdot 2\cdots i}.$$

Prvočíslo p delí čitateľ tohto zlomku, ale nedelí jeho menovateľ, lebo všetky čísla v menovateli sú menšie ako p. Z toho vyplýva, že $p \mid \binom{p}{i}$. □

Lema 2.1.2. *Nech p je prvočíslo. Potom*

$$a^p \equiv a \,(\mathrm{mod}\ p) \tag{2.1}$$

pre každé celé číslo a.

Dôkaz. Pre každé prvočíslo p máme $(-a)^p \equiv -a^p \,(\mathrm{mod}\ p)$, čiže stačí overiť platnosť kongruencie pre $a \geq 0$. Uvedená kongruencia zrejme platí pre $a = 0$, $a = 1$. Pre ostatné a ju môžeme dokázať indukciou vzhľadom na a.

Predpokladajme, že $a \geq 0$ a platí $a^p \equiv a \,(\mathrm{mod}\ p)$. Z binomickej vety máme $(a + 1)^p = \sum_{i=0}^{p} \binom{p}{i} a^i$. Použitím lemy 2.1.1 dostaneme

$$(a+1)^p \equiv a^p + 1 \equiv a + 1 \,(\mathrm{mod}\ p).$$

□

Veta 2.1.3 (Malá Fermatova veta). *Nech p je prvočíslo a nech a je celé číslo také, že $p \nmid a$. Potom*

$$a^{p-1} \equiv 1 \,(\mathrm{mod}\ p) \tag{2.2}$$

Dôkaz. Tvrdenie malej Fermatovej vety vyplýva z predchádzajúcej lemy a z toho, že $(a, p) = 1$. □

Poznamenajme, že obrátenie malej Fermatovej vety neplatí. Existuje nekonečne veľa zložených čísel, ktoré tiež spĺňajú kongruenciu (2.1) pre každé a. Tieto čísla sa nazývajú *absolútne pseudoprvočísla* alebo *Carmichaelove čísla* (pozri [5], [14], [18], [27]). Najmenšie Carmichaelove číslo je $561 = 3 \cdot 11 \cdot 17$.

Lema 2.1.4. *Nech $n \geq 2$ a $d \geq k \geq 1$ sú celé čísla. Potom $(n^k - 1) \mid (n^d - 1)$ práve vtedy, keď $k \mid d$.*

Dôkaz. $\boxed{\Rightarrow}$ Nech $d = mk + r$, $0 \leq r < k$. Potom dostaneme $n^d - 1 = n^{mk+r} - 1 = n^r(n^{mk} - 1) + n^r - 1$. Zrejme $n^k - 1 \mid n^{mk} - 1$ (lebo $n^{mk} - 1 = n^{k^m} - 1 = (n^k - 1)(n^{k^{m-1}} + \cdots + n^k + 1)$), čiže $n^k - 1 \mid n^r - 1$. Pretože $r < k$, dostali sme, že menšie z týchto dvoch nezáporných čísel je deliteľné väčším, čo môže nastať jedine v prípade $n^r - 1 = 0$, $r = 0$. Potom $d = mk$ a $k \mid d$.

Opačná implikácia je zrejmá. □

2.1.2 Čebyševove nerovnosti

V tejto časti bude platiť dohoda, že vždy keď vytvárame sumu alebo súčin a sčitujeme alebo násobíme všetky p z daného rozsahu, tak p predstavuje iba prvočísla. (Čiže ide o sumu alebo súčin len pre prvočísla patriace do tohto rozsahu.)

Pre ľubovoľné reálne číslo x definujeme *prvočíselnú funkciu* $\pi(x)$ ako počet prvočísel, ktoré sú menšie alebo rovné x. Funkcia π teda popisuje rozloženie prvočísel medzi prirodzenými číslami.

Jedným z najhlbších výsledkov teórie čísel je *prvočíselná veta,* ktorá vlastne dáva odhad pre rád funkcie $\pi(x)$. Táto veta hovorí, že

$$\lim_{x\to\infty} \frac{\pi(x)}{\frac{x}{\ln x}} = 1,$$

t.j. $\pi(x) \sim \frac{x}{\ln x}$.

Nám bude stačiť *Čebyševova veta*, ktorej platnosť ľahko vyplýva z platnosti prvočíselnej vety. Čebyševova veta hovorí, že existujú také reálne kladné konštanty c_1, c_2, že pre všetky $x \geq 2$ platí

$$c_1 \frac{x}{\ln x} \leq \pi(x) \leq c_2 \frac{x}{\ln x}.$$

Zvyšok tejto časti je venovaný hlavne dôkazu Čebyševovej vety.

Lema 2.1.5. *Pre každé reálne číslo $x \geq 2$ platí*

$$\prod_{p\leq x} p < 4^x,$$

pričom uvedený súčin berieme cez všetky prvočísla p nepresahujúce x.

Dôkaz. Najprv si všimnime, že stačí dokazovať lemu pre prirodzené čísla $n \geq 2$. Ak totiž lema platí pre každé prirodzené číslo, tak pre reálne číslo $x \geq 2$ dostaneme $\prod_{p\leq x} p = \prod_{p\leq \lfloor x\rfloor} p < 4^{\lfloor x\rfloor} \leq 4^x$.

Pre prirodzené čísla dokážeme tvrdenie lemy dokážeme matematickou indukciou, pričom budeme rozlišovať dva prípady - keď n je párne a keď n je nepárne. Pre $n = 1, 2$ tvrdenie platí. Predpokladajme teraz, že platí pre všetky čísla menšie ako n.

Ak $n = 2k$ pre nejaké prirodzené číslo $k > 1$, tak n nie je prvočíslo, čiže platí $\prod_{p\leq 2k} p = \prod_{p\leq 2k-1} p < 4^{2k-1} < 4^{2k}$.

Ak $n = 2k + 1$, tak platí

$$\prod_{p\leq 2k+1} p = \prod_{p\leq k} p \prod_{k<p\leq 2k+1} p < 4^{k+1} \prod_{k<p\leq 2k+1} p.$$

Kombinačné číslo

$$\binom{2k+1}{k+1} = \frac{(2k+1)\cdot(2k)\cdots(k+2)}{1\cdot 2\cdots k}$$

je deliteľné každým prvočíslom p, pre ktoré $k+1 < p \leq 2k+1$. (Takéto prvočísla delia čitateľ ale nedelia menovateľ uvedeného zlomku.) Preto platí $\prod_{k<p\leq 2k+1} p \leq \binom{2k+1}{k+1}$. Tento binomický koeficient môžeme ľahko odhadnúť na základe nerovnosti

$$2^{2k+1} > \binom{2k+1}{k+1} + \binom{2k+1}{k} = 2\binom{2k+1}{k+1},$$

z ktorej dostaneme

$$\prod_{k<p\leq 2k+1} p \leq \binom{2k+1}{k+1} < 2^{2k} = 4^k.$$

Celkovo teda dostávame, že $\prod_{p\leq 2k+1} p < 4^{k+1}\cdot 4^k = 4^{2k+1}$. □

Veta 2.1.6. *Pre každé dostatočne veľké číslo n platí*

$$\pi(n) \leq \frac{5n}{\lg n}.$$

Dôkaz. V dôkaze odhadneme zhora aj zdola výraz $\sum_{p\leq n} \lg p$.

$$\sum_{p\leq n} \lg p \geq \sum_{\sqrt{n}<p\leq n} \lg p \geq \sum_{\sqrt{n}<p\leq n} \lg \sqrt{n} = (\pi(n) - \pi(\sqrt{n})) \lg \sqrt{n}$$

Z lemy 2.1.5 dostaneme

$$\sum_{p\leq n} \lg p = \lg \left(\prod_{p\leq n} p \right) \leq 2n.$$

Spojením týchto dvoch nerovností dostaneme $\pi(n) \leq \frac{4n}{\lg n} + \pi(\sqrt{n}) \leq \frac{4n}{\lg n} + \sqrt{n}$. Pre dostatočne veľké n platí $\sqrt{n} \leq \frac{n}{\lg n}$, z čoho vyplýva

$$\pi(n) \leq \frac{5n}{\lg n}.$$

□

Pre ľubovoľné kladné číslo n označme $d_n = [1, 2, \ldots, n]$ najmenší spoločný násobok prvých n prirodzených čísel. Nasledujúci dôkaz dolného odhadu pre $\pi(n)$ je z článku [29].

Lema 2.1.7. *Pre každé kladné číslo n platí $d_n \geq 2^{n-2}$.*

Dôkaz. Označme $I := \int_0^1 x^m(1-x)^m dx$. Pre každé $x \in (0,1)$ platí $0 < x(1-x) = \frac{1}{4} - (x - \frac{1}{2})^2 \leq \frac{1}{4}$, z čoho vyplýva $0 < I \leq \frac{1}{4^m}$.

Súčasne platí $I = \int_0^1 \sum_{k=0}^{m} x^{m+k} \binom{m}{k} x^k (-1)^{m-k} dx = \sum_{k=0}^{m} \binom{m}{k} (-1)^{m-k} \int_0^1 x^{m+k} dx = \sum_{k=0}^{m} (-1)^{m-k} \binom{m}{k} \frac{1}{m+k+1}$.

Po úprave na spoločného menovateľ dostaneme zlomok, ktorého menovateľ je najviac d_{2m+1}, pretože d_{2m+1} je najmenší spoločný násobok všetkých zlomkov, ktoré vystupujú v súčte. Môžeme teda uvedený integrál vyjadriť v tvare $I = \frac{A}{d_{2m+1}}$, kde $A > 0$ je prirodzené číslo. Potom platí pre $n = 2m + 1$

$$d_n = d_{2m+1} \geq 4^m = 2^{n-1}.$$

Ak n je párne, tak platí $d_n \geq d_{n-1} = 2^{n-2}$. □

Veta 2.1.8. *Pre každé kladné číslo n platí*

$$\pi(n) \geq \frac{n-2}{\lg n}.$$

Dôkaz. Ak $p_1, \ldots, p_k$ sú všetky prvočísla, ktoré sú menšie alebo rovné n. Každé číslo $m = 1, \ldots, n$ má rozklad tvaru

$$m = \prod_{i=1}^{k} p_i^{s_{m_i}},$$

kde $s_{m_i} \geq 0$ pre všetky $i = 1, \ldots, k$. Potom najmenší spoločný násobok d_n čísel $1, 2, \ldots, n$ má tvar

$$d_n = \prod_{i=1}^{k} p_i^{\max\{s_{1_i}, \ldots, s_{n_i}\}}.$$

Zrejme platí $p_i^{\max\{s_{1_i}, \ldots, s_{n_i}\}} \leq n$ pre každé $i = 1, \ldots, k$. Z toho vyplýva, že $d(n) \leq n^k = n^{\pi(n)}$. Z toho dostaneme podľa lemy 2.1.7 $\pi(n) = \frac{\lg d(n)}{\lg n} \geq \frac{n-2}{\lg n}$. □

Ďalšou dôležitou funkciou v teórii čísel je *Čebyševova funkcia* $\vartheta(x)$, ktorá je definovaná ako

$$\vartheta(x) = \sum_{p \leq x} \ln p.$$

Podľa dohody na začiatku tejto časti uvedenú sumu berieme len cez prvočísla z daného rozsahu.

Nasledujúca veta zachytáva vzťah medzi funkciami $\pi(x)$ a $\vartheta(x)$.

Veta 2.1.9.

$$\pi(x) \sim \frac{\vartheta(x)}{\ln x}$$

Dôkaz. Zrejme $\vartheta(x) = \sum\limits_{p \leq x} \lg p \leq \sum\limits_{p \leq x} \lg x = \pi(x) \lg x$. Z toho dostaneme, že

$$\frac{\vartheta(x)}{\pi(x) \ln x} \leq 1.$$

Majme teraz $x \geq 2$ a $0 < \varepsilon < 1$. Potom platí

$$\vartheta(x) \geq \sum_{x^{1-\varepsilon} < p \leq x} \ln p \geq (1-\varepsilon) \ln x (\pi(x) - \pi(x^{1-\varepsilon})) \geq (1-\varepsilon) \ln x (\pi(x) - x^{1-\varepsilon}).$$

Z toho dostaneme

$$\frac{\vartheta(x)}{\ln x \pi(x)} \geq (1-\varepsilon) \left(1 - \frac{x^{1-\varepsilon}}{\pi(x)}\right) \geq (1-\varepsilon) \left(1 - \frac{x^{1-\varepsilon} \ln x}{5x \ln 2}\right).$$

Ak urobíme limitu pre x idúce do nekonečna, tak máme

$$\lim_{n \to \infty} \frac{\vartheta(x)}{\ln x \pi(x)} \geq 1 - \varepsilon.$$

Pretože ε môžeme zvoliť ľubovoľne malé, platí potom

$$\lim_{n \to \infty} \frac{\vartheta(x)}{\ln x \pi(x)} = 1.$$

□

Dôsledok 2.1.10. *Existujú také reálne konštanty $A, B > 0$, že pre všetky $x \geq 2$ platí*

$$Ax \leq \vartheta(x) = \sum_{p \leq x} \ln p \leq Bx.$$

2.2 Konečné polia

Lema 2.2.1. *Nech G je konečná grupa, $1 \in G$ je jej neutrálny prvok a $|G| = n$. Potom $a^n = 1$ pre každý prvok $a \in G$.*

Lemu 2.2.1 môžeme použiť na iný dôkaz malej Fermatovej vety. Stačí si uvedomiť, že $\mathbb{Z}_p$ je pole, a teda $\mathbb{Z}_p \setminus \{0\}$ s násobením je $(p-1)$-prvková grupa.

Ak A je okruh, tak okruh $A[x]$ polynómov nad A v neurčitej x obsahuje polynómy s koeficientami z A, pričom sčitovanie a násobenie sa definuje prirodzeným spôsobom.

Okruh $A[x]$ je obor integrity práve vtedy, keď A je obor integrity. Ak F je pole, tak v $F[x]$ môžeme deliť so zvyškom (je to tzv. euklidovský okruh), čiže $F[x]$ je potom aj okruh hlavných ideálov.

Nech F je pole a $F[x]$ je okruh polynómov nad poľom F. Polynóm $f(x)$ je *ireducibilný (nerozložiteľný)* v okruhu $F[x]$, ak neexistujú polynómy $g(x)$ a $h(x)$, také, že $f(x) = g(x).h(x)$ a žiadny z polynómov $g(x)$, $h(x)$ nie je stupňa 0 (t.j. nie je konštantný polynóm).

Ak p je prvočíslo, tak okruh $\mathbb{Z}_p$ so sčitovaním a násobením modulo p tvorí pole. Ak $h(x)$ je ireducibilný polynóm stupňa d, tak aj $F = \mathbb{Z}_p[x]/(h(x))$ je pole. Pole F má p^d-prvkov a obsahuje práve polynómy nad $\mathbb{Z}_p[x]$ stupňa menšieho ako d. Násobenie v F je definované modulo $h(x)$. (Presnejšie povedané, triedy rozkladu okruhu $\mathbb{Z}_p[x]$ podľa hlavného ideálu generovaného polynómom $h(x)$ môžeme stotožniť s polynómami stupňa menšieho ako d - reprezentantmi jednotlivých tried.)

Pre polynómy nad poľom F platí veta o rozklade na ireducibilné polynómy:

Veta 2.2.2 ([21, Veta 5.4.1]). *Ak $f(x) = a_n x^n + a_{n-1}x^{n-1} + \ldots + a_0 \in F[x]$ je polynóm stupňa n nad poľom F, tak existujú normované ireducibilné polynómy $p_1(x), \ldots, p_m(x)$ nad poľom F také, že platí*

$$f(x) = a_n p_1(x) \ldots p_m(x).$$

Tento rozklad je určený jednoznačne až na poradie činiteľov.

Prvok $c \in F$ nazývame *koreňom* polynómu $f(x) = a_n x^n + a_{n-1}x^{n-1} + \ldots + a_0$ nad poľom, ak platí $f(c) = a_n c^n + a_{n-1}c^{n-1} + \ldots + a_0 = 0$. Prvok c je koreňom $f(x)$ práve vtedy, keď polynóm $x - c$ delí polynóm $f(x)$ v $F[x]$. Ak $(x-c)^k \mid f(x)$, pričom k je najväčšie takéto prirodzené číslo, tak hovoríme, že c je *k-násobným koreňom* polynómu $f(x)$.

Ako dôsledok predchádzajúcej vety dostávame (pozri vetu 5.5.1 v [21] a za ňou nasledujúci dôsledok):

Veta 2.2.3. *Ak $f(x) = a_n x^n + a_{n-1}x^{n-1} + \ldots + a_0$ je polynóm stupňa n nad poľom F a F_1 je nadpole poľa F, tak $f(x)$ má najviac n koreňov v poli F_1 vrátane násobnosti (t.j. každý koreň započítame toľkokrát, koľko je jeho násobnosť).*

Veta 2.2.4. *Multiplikatívna grupa $F^* = (F \setminus \{0\}, \cdot)$ ľubovoľného konečného poľa F je cyklická.*

Dôkaz. Nech q označuje počet prvkov poľa F, počet prvkov multiplikatívnej grupy F^* je potom $q-1$. Nech $q-1$ má kanonický rozklad $p_1^{s_1} \dots p_r^{s_r}$. Najprv ukážeme, že pre každé $i = 1, \dots, r$ existuje v F prvok b_i rádu $p_i^{s_i}$. Generátor grupy F^* bude súčin týchto prvkov $b := b_1 \dots b_r$.

Polynóm $x^{\frac{q-1}{p_i}}$ má v F podľa vety 2.2.3 najviac $\frac{q-1}{p_i}$ koreňov. Teda existuje $a_i \in F$ také, že $a_i^{\frac{q-1}{p_i}} \neq 1$. Položme $b_i := a_i^{(q-1)/p_i^{s_i}}$. Multiplikatívny rád prvku b_i je $p_i^{s_i}$, pretože $b_i^{p_i^{s_i}} = 1$, ale $b_i^{p_i^{s_i-1}} = a_i^{\frac{q-1}{p_i}} \neq 1$.

Položme teraz $b := b_1 \dots b_r$. Označme multiplikatívny rád b ako e. Tvrdíme, že $e = q-1$. Zrejme $e \mid q-1$. Predpokladajme, že by pre niektoré $i = 1, \dots, r$ platilo $p_i^{s_i} \nmid e$. Potom $e \mid \frac{q-1}{p_i} =: f$. Pre $j \neq i$ zrejme platí $b_j^f = 1$, z čoho dostaneme $b^f = b_1^f \dots b_r^f = b_i^f \neq 1$, ale súčasne platí $e \mid f$ a $b^e = 1$, čo je spor. □

Lema 2.2.5. *Nech p je prvočíslo a $f(x)$ je polynóm nad poľom $\mathbb{Z}_p$. Potom v $\mathbb{Z}_p[x]$ platí rovnosť*

$$f(x^p) = (f(x))^p.$$

Dôkaz. Budeme postupovať indukciou vzhľadom na stupeň n polynómu $f(x)$. Ak $n = 0$, tak tvrdenie vyplýva z lemy 2.1.2.

Predpokladajme, že $n \leq 1$ a uvedené tvrdenie platí pre všetky polynómy stupňa menšieho ako n. Potom $f(x) = ax^n + g(x)$, kde $a \in \mathbb{Z}_p$ a $g(x)$ je polynóm stupňa $n-1$. Použitím binomickej vety dostaneme podobne ako v dôkaze lemy 2.1.2, že $(f(x))^p = (g(x))^p + a^p x^{pn}$. Využitím toho, že $a^p \equiv a \pmod p$ a $(g(x))^p = g(x^p)$ (indukčný predpoklad) dostávame

$$(f(x))^p = g(x^p) + a(x^p)^n = f(x^p).$$

□

Veta 2.2.6. *Nech p a r sú dve navzájom rôzne prvočísla a s je multiplikatívny rád p modulo r. (Teda s je najmenšie také prirodzené číslo, že $p^s \equiv 1 \pmod r$.) Každý ireducibilný polynóm $h(x)$ v $\mathbb{Z}_p[x]$, ktorý delí $\frac{x^r-1}{x-1}$ má stupeň s. Navyše platí, že rád prvku x v multiplikatívnej grupe F^* poľa $F = \mathbb{Z}_p[x]/(h(x))$ sa rovná r.*

Dôkaz. Nech $h(x)$ je ireducibilný polynóm v $\mathbb{Z}_p[x]$, ktorý delí $\frac{x^r-1}{x-1}$ a nech k je stupeň $h(x)$. Ukážeme, že $k \mid s$ a $s \mid k$. Keďže k aj s sú prirodzené čísla, vyplýva z toho už $k = s$.

Keďže $h(x)$ je ireducibilný polynóm, $\mathbb{Z}_p[x]$ je p^k-prvkové pole. Nech $g(x)$ je generátor multiplikatívnej grupy poľa $\mathbb{Z}_p[x]/(h(x))$. (Podľa vety 2.2.4 je táto grupa cyklická, čiže má generátor.) Lema 2.2.5 hovorí, že $(g(x))^p = g(x^p)$ v $\mathbb{Z}_p[x]$. Viacnásobným použitím tejto rovnosti dostaneme postupne $(g(x))^{p^2} = g(x^{p^2})$, $(g(x))^{p^3} = g(x^{p^3})$ až

$$(g(x))^{p^s} = g(x^{p^s}) \tag{2.3}$$

v $\mathbb{Z}_p[x]$. Z kongruencie $p^s \equiv 1 \pmod r$ vyplýva, že existuje prirodzené číslo m také, že $p^s = 1 + mr$. Nech $f(x)$ je polynóm v $\mathbb{Z}_p[x]$, pre ktorý $f(x)h(x) = x^r - 1$. Potom v

$\mathbb{Z}_p[x]$ platí rovnosť

$$x^{p^s} = xx^{mr} = x(1 + f(x)h(x))^m = x \mod h(x).$$

Spolu s rovnosťou (2.3) máme potom

$$(g(x))^{p^s} = g(x) \mod h(x).$$

Keďže $g(x)$ patrí do multiplikatívnej grupy poľa $\mathbb{Z}_p[x]/(h(x))$, má inverzný prvok vzhľadom na násobenie. Keď predošlú rovnosť vynásobíme týmto prvkom, dostaneme

$$(g(x))^{p^s-1} = 1 \mod h(x).$$

Rád polynómu $g(x)$ v $(\mathbb{Z}_p[x]/(h(x)))^*$ je $p^k - 1$, preto $(p^k - 1) \mid (p^s - 1)$. Podľa lemy 2.1.4 teda platí $k \mid s$.

Keďže $h(x)$ delí $x^r - 1$ v $\mathbb{Z}_p[x]$, máme $x^r = 1 \mod h(x)$. Súčasne $x \neq 1 \mod h(x)$. (Formálnym derivovaním $x^r - 1$ zistíme, že 1 nie je viacnásobný koreň $x^r - 1$, preto nemôže byť koreňom polynómu $\frac{x^r-1}{x-1}$, a teda ani koreňom jeho deliteľa $h(x)$.) Z toho, že r je prvočíslo, už potom vyplýva, že rád polynómu x v $\mathbb{Z}_p[x]/(h(x))$ sa rovná r.

Súčasne vieme, že multiplikatívna grupa poľa $\mathbb{Z}_p[x]/(h(x))$ má $p^k - 1$ prvkov, a teda rád ľubovoľného prvku delí $p^k - 1$. Dostávame, $r \mid (p^k - 1)$, čiže $p^k \equiv 1 \pmod r$. Pretože multiplikatívny rád p modulo r je s, vyplýva z poslednej kongruencie $s \mid k$. □

2.3 Zložitosť algoritmov

Cieľom tejto kapitoly je uviesť zložitosť niektorých algoritmov, ktoré využijeme ako pomocné pri implementovaní AKS-algoritmu. Keďže chceme, aby náš výklad bol elementárny, uvedieme pre základné operácie s veľkými celými číslami a s polynómami, ktoré majú celočíselné koeficienty najprv jednoduché algoritmy, ktorých zložitosť možno ľahko dokázať. Súčasne však uvedieme aj odhady na zložitosť oveľa rýchlejších spôsobov na implementáciu týchto operácií pomocou rýchlej Fourierovej transformácie (FFT). V ďalšom budeme pre každý algoritmus uvádzať dva odhady - v závislosti od toho, či použijeme rýchlejšie alebo pomalšie násobenie a delenie veľkých čísel resp. polynómov.

2.3.1 Celočíselná a modulárna aritmetika

Lema 2.3.1. *Ak a, b sú prirodzené čísla také, že platí $1 \leq a \leq b \leq n$, kde $n \in \mathbb{N}$, tak existujú algoritmy na výpočet ich súčinu ab v čase $O(\lg^2 n)$ a ich celočíselného podielu $\lfloor \frac{b}{a} \rfloor$ a zvyšku $b \mod a$ v čase $O(\lg n)$.*

Dôkaz lemy 2.3.1 len naznačíme. V prípade násobenia veľkých celých čísel, ktoré sú reprezentované ako postupnosť cifier môžeme použiť algoritmus, ktorý bežne používame pri násobení čísel „na papieri“. Takto vlastne vypočítame súčiny jedného z čísel s ciframi druhého čísla a tieto medzivýsledky - pričom sú patrične posunuté - potom sčitujeme. Pre úplný popis algoritmu by ešte bolo treba uviesť ako sa spracuje prenos, ktorý vznik pri sčitovaní cifier na tomto mieste. Je však zrejmé, že zložitosť algoritmu je daná rozmermi „útvaru“, ktorého cifry sčitujeme, čo je v našom prípade $O(\lg^2 n)$.

Pri výpočte celočíselného podielu a zvyšku predpokladajme, že číslo je reprezentované v binárnej sústave. Porovnaním dĺžok binárnych zápisov čísel a a b (a prípadným posunom o 1 cifru) môžeme nájsť prvú cifru celočíselného podielu (resp. najbližší možný násobok čísla b mocninou dvojky, ktorý je menší alebo rovný a). Označme tento násobok číslom m. V jednotlivých krokoch vždy zmenšujeme tento násobok na polovicu a v prípade, že je menší ako obsah premennej a, premennú a zmenšíme o m a vieme, že v binárnom rozvoji podielu bude na príslušnom mieste jednotka. Keď dospejeme do situácie, že a je menšie ako b, tak v premennej a je číslo $\lfloor \frac{a}{b} \rfloor$. Zrejme celý tento proces trvá $O(\lg n)$.

Bez dôkazu budeme používať lemu 2.3.2, ktorá hovorí, že základné operácie, konkrétne násobenie a delenie so zvyškom možno spočítať v čase $\tilde{O}(\lg n)$. Analogické tvrdenia pre polynómy sú sformulované v leme 2.3.6. Ide vlastne o tie isté odhady, keď si uvedomíme, že na zápis polynómu stupňa d potrebujeme d-krát viac bitov. Tieto výsledky možno nájsť napríklad v [9], [14], [22].

Lema 2.3.2. *Ak a, b sú prirodzené čísla také, že platí $1 \leq a \leq b \leq n$, kde $n \in \mathbb{N}$, tak existujú algoritmy na výpočet ich súčinu ab, ich celočíselného podielu $\lfloor \frac{b}{a} \rfloor$ a zvyšku $b \bmod a$ v čase $\tilde{O}(\lg n)$.*

Lema 2.3.3. *Nech $a \geq 2$ a $b \geq 1$ sú celé čísla. Potom existuje algoritmus, ktorý počíta mocninu $n = a^b$ v čase $O(\lg^2 n)$ (pomocou FFT) resp. v čase $O(\lg b \lg^2 n)$ (pri pomalšom násobení).*

Dôkaz. Mocninu a^b môžeme rátať nasledujúcim algoritmom:

```
int Pow(int a, int b) {
  int x,y,z;

  x=a; y=b; z=1;
  while (y!=0) {
    if (y%2) {//y je nepárne
      y--; z=z*x;
    } else {//y je párne
      y=y/2; x=x*x;
    }
  }
}
```

Správnosť algoritmu vyplýva z toho, že vždy po skončení `while` cyklu platí invariant $zx^y = a^b$.

Počet opakovaní `while` cyklu, ktoré je potrebné urobiť je $O(\lg b)$. Číslo $c = \lfloor \lg b \rfloor$ je najmenšie celé číslo, pre ktoré platí $b < 2^{c+1}$. Všimnime si, že čísla, ktoré násobíme v jednotlivých iteráciach majú veľkosť nanajvýš $a, a^2, \ldots, a^{2^{c-1}}, a^{2^c}$. Podľa lemy 2.3.2 teda jednotlivé iterácie budú trvať najviac $(1 + 2 + \cdots + 2^{c-1} + 2^c)O(\lg^2 a) = O(2^c \lg^2 a) = O(b \lg^2 a)$. Z toho dostaneme odhad pre celkovú časovú zložitosť algoritmu $O(b^2 \lg^2 a) = O(\lg^2 n)$.

Ak použijeme pomalšie násobenie, tak dostaneme zložitosť pre jednu iteráciu $(1 + 2^2 + \cdots + 2^{2c})O(\lg^a) = O(b^2 \lg^2 a) = O(\lg b \lg^2 n)$. □

Lema 2.3.4. *Nech $a \geq 2$, $b \geq 1$ a $r \geq 2$ sú celé čísla. Potom existuje algoritmus, ktorý ráta hodnotu $a^b \mod r$ v čase $\tilde{O}(\lg n + \lg b \lg r)$ (pri FFT) resp. $O(\lg n + \lg b \lg^2 r)$ (pomalšie násobenie), kde $n = \max\{a, r\}$.*

Dôkaz. Použijeme úplne rovnaký algoritmus ako v predchádzajúcej leme s tým rozdielom, že x inicializujeme ako $a \mod r$ a v každom kroku počítame v závislosti od parity y buď nové x ako $x^2 \mod r$ alebo z ako $zx \mod r$.

Inicializácia podľa lemy 2.3.2 trvá $\tilde{O}(\lg n)$, samotný výpočet prebieha v $O(\lg b)$ iteráciach z ktorých každá trvá $\tilde{O}(\lg r)$.

Pri pomalšom násobení trvá z lemy 2.3.1 dostaneme inicializáciu v čase $O(\lg n)$ a trvanie jednej iterácie $O(\lg^2 r)$. □

2.3.2 Počítanie s polynómami

Algoritmy použité v leme 2.3.5 sú vlastne bežné algoritmy, ktoré obvykle používame pri počítaní s polynómami. Algoritmy v leme 2.3.6 sú analogické k algoritmom pre počítanie s celými číslami, ktoré sú uvedené v leme 2.3.2. Opäť platí dohoda, že budeme uvádzať časovú zložitosť pri použití rýchlejšieho aj pomalšieho násobenia.

Lema 2.3.5. *Sčítanie, násobenie a delenie dvoch polynómov stupňa d s koeficientami veľkosti najviac e možno vykonať v čase $O(d^2 \lg^2 e)$.*

Lema 2.3.6. *Sčítanie, násobenie a delenie dvoch polynómov stupňa d s koeficientami veľkosti najviac e možno vykonať v čase $\tilde{O}(d \lg e)$.*

Lema 2.3.7. *Nech $n, r, a \in \mathbb{N}$, $2 \leq r < n$ a $1 \leq a < n$. Všetkých r koeficientov polynómu $(x - a)^n \mod (x^r - 1)$ v $\mathbb{Z}_n[x]$ možno vyrátať v čase $\tilde{O}(r \lg^2 n)$ (FFT), resp. $O(r^2 \lg^3 n)$ (pomalšie násobenie).*

Dôkaz. Použijeme podobný algoritmus ako v leme 2.3.3. Uvedieme symbolický zápis algoritmu. Predpokladáme, že všetky operácie s koeficientami uvedených polynómov sa robia v $\mathbb{Z}_n$, t.j. modulo n.

```
polynom Koef(int n) {
  int y;
  polynom f,g,h;

  f(x)=1; g(x)=x-a; y=n;
  while (y!=0) {
    if (y%2) {//y je nepárne
      y--; h(x)=f(x)*g(x);
      g(x) = h(x) mod (x^r - 1);
    } else {//y je párne
      y=y/2; h(x)=g(x)*g(x);
      f(x) = h(x) mod (x^r - 1);
    }
  }
  return(f(x));
}
```

Zdôvodnenie správnosti je presne také isté ako v leme 2.3.3 - platí invariant $f(x)g(x)^y = (x-a)^n \mod (x^r-1)$ v $\mathbb{Z}_n[x]$.

Keďže stupeň použitých polynómov nepresiahne $2r$ a všetky ich koeficienty majú veľkosť menej než n, časová zložitosť jednej iterácie bude podľa lemy 2.3.6 $O\tilde{}(r \lg n)$. (To, že namiesto obvyklého násobenia a sčitovania sa pri operáciach s polynómami bude používať modulárna aritmetika časovú zložitosť nezhorší.) Pri pomalšom násobení (2.3.5) máme časovú zložitosť pre jednu iteráciu $O(r^2 \lg^2 n)$.

Počet iterácií je $O(\lg n)$, dostávame teda zložitosť $O\tilde{}(r \lg^2 n)$, resp. $O(r^2 \lg^3 n)$. □

Inou alternatívou ako realizovať výpočty s polynómami je použiť veľké celé čísla. Polynóm môžeme chápať ako zápis čísla v pozičnej sústave, pričom „cifry“ môžu byť ľubovoľne veľké (presnejšie povedané ľubovoľné prvky poľa, ktorého koeficienty vystupujú v polynóme). T.j. rozdiel v algoritmoch na sčitovanie, násobenie a delenie oproti celým číslam a modulárnej aritmetike je ten, že nikdy nedochádza k prenosu do vyššieho rádu (pozri [22]).

2.3.3 Prvočísla

Nasledujúci algoritmus zisťuje či n je prvočíslo. Ide o najjednoduchší algoritmus na zisťovanie prvočíselnosti: n jednoducho delíme číslami menšími než je $\lfloor\sqrt{n}\rfloor$. Pretože jediné párne prvočíslo je 2 stačí testovať deliteľnosť dvojkou a nepárnymi číslami.

```
bool Is_Prime(int n) {
  int p,s;

  if (n<2) return(FALSE);
  if (n==2)
    return(TRUE);
  if (n%2==0)
    return(FALSE);//párne číslo
  p=3; s=p*p; //invariant: s=p^2
  while (s<=n) {
    if (n%p==0)
      return(FALSE);
    s+=4*p+4;
    p+=2;
  }
  return(TRUE);
}
```

Lema 2.3.8. *Algoritmus* `Is_Prime` *rozhodne či n je prvočíslo v čase $O\tilde{}(\sqrt{n}\lg n)$ (FFT), resp. $O(\sqrt{n}\lg^2 n)$ (pomalšie násobenie).*

Dôkaz. Správnosť algoritmu `Is_Prime` vyplýva z toho, že každé zložené číslo je deliteľné nejakým číslom $p \le \lfloor\sqrt{n}\rfloor$. To znamená, že p spĺňa nerovnosť $p^2 \le n$. Počas celého výpočtu platí invariant $s = p^2$ a pred každým opakovaním `while` cyklu testujeme, či $p^2 = s \le n$.

Počet opakovaní `while` cyklu je $O(\sqrt{n})$ a každé opakovanie trvá $O\tilde{}(\lg n)$, resp. $O(\lg^2 n)$. □

V [22] možno nájsť niektoré vylepšenia algoritmov `Is_Prime()`, ktoré síce nezlepšia asymptotický odhad zložitosti algoritmov, v praxi však niekoľkonásobne urýchlia výpočet. Ide napríklad o použitie tabuľky malých prvočísel (menších ako 100), vynechanie delenia všetkých násobkov 3 (prípadne 5,7,...) podobne ako sme tu vynechali delenie párnymi číslami.

2.3.4 Testovanie, či n je mocninou celého čísla

Lema 2.3.9. *Nech $n \geq 2$ je celé číslo. Potom existuje algoritmus, ktorý v čase $O(\lg^3 n \lg\lg n)$ (resp. $O(\lg^4 n \lg\lg n)$ pri použití pomalšieho násobenia) rozhodne, či existujú celé čísla $a \geq 2$ a $b \geq 2$ také, že $n = a^b$.*

Dôkaz. Ak $n = a^b$, tak zrejme platí $2^b \leq n$ a $b \leq \lfloor \lg n \rfloor$. Stačí teda vyskúšať možných kandidátov na číslo b od 2 po $\lfloor \lg n \rfloor$.

Ak už máme vybraného konkrétneho kandidáta b, tak chceme zistiť, či existuje a také, že $n = a^b$. Označme $c := \lfloor \lg n \rfloor$, zrejme platí $2^c \leq n < 2^{c+1}$. Z rovnosti $n = a^b$ vyplýva $2^{\frac{c}{b}} \leq a < 2^{\frac{c+1}{b}}$ a $2^{\lfloor \frac{c}{b} \rfloor} \leq a < 2^{\lceil \frac{c+1}{b} \rceil}$.

Hodnota a leží teda v intervale $u := 2^{\lfloor \frac{c}{b} \rfloor}$ až $v := 2^{\lceil \frac{c+1}{b} \rceil}$ a môžeme ju nájsť binárnym prehľadávaním.

1. Ak $u \geq v$, tak n nie je mocninou čísla a,
2. $w = \lfloor \frac{u+v}{2} \rfloor$,
3. $z = w^b$,
4. (a) ak $z = n$, tak $n = w^b$ a našli zistili sme, že číslo n je tvaru a^b,
   (b) ak $z < n$, tak priradíme $u = w + 1$ a pokračujeme v binárnom prehľadávaní (platí $w + 1 \leq a < v$),
   (c) ak $z > n$, tak priradíme $v = w$ a pokračujeme v binárnom prehľadávaní (platí $u \leq a < w$).

Správnosť algoritmu vyplýva z toho, že ak $n = a^b$, tak v priebehu testovania kandidáta b stále platí invariant $u \leq a < v$.

Najprv odhadnime časovú zložitosť pre testovanie jedného konkrétneho kandidáta b. Počet opakovaní binárneho vyhľadávania je $O(\frac{c}{b}) = O(\frac{\lg n}{b})$. V priebehu binárneho vyhľadávania potrebujeme vyrátať hodnotu w^b. Keďže platí $w^b \leq 2^{\lceil \frac{c+1}{b} \rceil e} \leq 2^{e+\lg n} \leq 2^{2\lg n} = O(n^2)$, tak výpočet hodnoty w^b trvá podľa lemy 2.3.3 nanajvýš $O(\lg^2 n)$. Celková zložitosť časti algoritmu, v ktorej testujeme b ako kandidáta na exponent je teda $O(\frac{\lg^3 n}{b})$.

Pretože postupne skúšame hodnoty b od 2 po $\lfloor \lg n \rfloor$, dostávame celkovú zložitosť $O\left(\sum_{e=1}^{\lfloor \lg n \rfloor} \frac{\lg^3 n}{b}\right) = O(\lg^3 n \lg\lg n)$.

Pri použití pomalšieho násobenia dostaneme trvanie jednej iterácie $O(\frac{\lg b \lg^3 n}{b}) = O(\frac{\lg^4 n}{b})$, z čoho dostaneme celkovú zložitosť $O(\lg^4 n \lg\lg n)$ □

Poznamenajme, že v článku [8] je uvedený algoritmus na testovanie, či n je mocnina prirodzeného čísla v čase $O(\lg^{1+\varepsilon} n)$ pre ľubovoľné $\varepsilon > 0$. Pre naše účely však stačí

aj algoritmus, ktorý sme uviedli, pretože celková časová zložitosť AKS-algoritmu je vyššia, než zložitosť kroku, v ktorom sa použije tento test.

2.3.5 Prehľad zložitosti algoritmov

Algoritmus	Vstup	Obvyklé násobenie	Násobenie pomocou FFT
násobenie celých čísel	$a, b \leq n$	$O(\lg^2 n)$	$O\tilde{}(\lg n)$
delenie celých čísel	$a, b \leq n$	$O(\lg n)$	$O\tilde{}(\lg n)$
mocnina $n = a^b$	a, b	$O(\lg b \lg^2 n)$	$O(\lg^2 n)$
násobenie a delenie polynómov	$\mathrm{st} f(x), \mathrm{st} g(x) \leq d$, koeficienty veľkosti e	$O(d^2 \lg^2 e)$	$O\tilde{}(d \lg e)$
$(x-a)^n \mod (x^r - 1)$ v $\mathbb{Z}_n[x]$	a, r, n	$O(r^2 \lg^3 n)$	$O\tilde{}(r \lg^2 n)$
Eratostenovo sito	n	$O(\sqrt{n} \lg^2 n)$	$O\tilde{}(\sqrt{n} \lg^2 n)$
netriviálna mocnina	n	$O(\lg^4 n \lg\lg n)$	$O(\lg^3 n \lg\lg n)$

Kapitola 3

Správnosť AKS-algoritmu

Obsahom tejto časti je odvodenie vety, ktorá zaručuje správnosť AKS-algoritmu. Podrobný popis AKS-algoritmu spolu s odhadom jeho zložitosti uvedieme v ďalšej kapitole.

3.1 Polynomická identita na testovanie prvočíselnosti

Základom AKS-algoritmu je použitie identity dokázanej v nasledujúcej vete. Veta 3.1.1 je zovšeobecním malej Fermatovej vety.

Veta 3.1.1. *Nech $n \geq 2$ a $a \geq 0$ sú celé čísla.*

(i) Ak n je prvočíslo, tak v okruhu $\mathbb{Z}_n[x]$ platí rovnosť

$$(x+a)^n = x^n + a.$$

(ii) Ak $(a,n) = 1$ a n nie je prvočíslo tak

$$(x+a)^n \neq x^n + a.$$

v okruhu polynómov $\mathbb{Z}_n[x]$.

Dôkaz. (i) Z binomickej vety dostaneme

$$(x+a)^n = \sum_{i=0}^{n} \binom{n}{i} x^i a^{n-i}.$$

Ak n je prvočíslo, tak podľa lemy 2.1.1 $\binom{n}{i} \equiv 0 \pmod{n}$ pre $i = 1, \ldots, n-1$ a podľa lemy 2.1.2 $a^n \equiv a \pmod{n}$. Preto v $\mathbb{Z}_n[x]$ platí

$$(x+a)^n = x^n + a^n = x^n + a.$$

(ii) Nech $(a,n) = 1$ a q je niektorý prvočiniteľ čísla n. Ako k označíme mocninu prvočísla q v kanonickom rozklade čísla n (teda k je také číslo, že $q^k \mid n$, ale $q^{k-1} \nmid n$). Potom dostaneme, že

$$\binom{n}{q} = \frac{n(n-1)\cdots(n-q+1)}{q!} = cq^{k-1},$$

kde c je prirodzené číslo také, že $(q, c) = 1$. Vyplýva to z toho, že spomedzi čísel v čitateli sa q vyskytuje iba v rozklade čísla n v k-tej mocnine, a v menovateli sa vyskytuje tiež práve raz, ale v prvej mocnine. Dostávame teda $q^k \nmid \binom{n}{q}$ a $q^{k-1} \mid \binom{n}{q}$.

Z predpokladu $(a, n) = 1$ vyplýva, že súčasne platí $(q^k, a^q) = 1$, a teda $q^k \nmid \binom{n}{q} a^q$. Potom takisto $n \nmid \binom{n}{q} a^q$. Teda q-ty koeficient v binomickom rozvoji $(x + a)^n$ v okruhu $\mathbb{Z}_n$ je nenulový a polynómy $(x + a)^n$ a $x^n + a^n$ sa nerovnajú v $\mathbb{Z}_n[x]$. □

3.2 Identita použitá v AKS-algoritme

Výhoda vety 3.1.1 oproti malej Fermatovej vete je v tom, že na rozdiel od nej nám poskytuje kritérium na zistenie, či n je alebo nie je prvočíslo. Ak by sme však na overenie, či n je prvočíslo túto vetu, získali by sme algoritmus, ktorý beží v exponenciálnom čase, pretože by sme museli vyrátať n koeficientov polynómu $(x - a)^n$. Autorom AKS-algoritmu sa podarilo nájsť vylepšenie tejto vety. Jeho dôkaz bude obsahom tejto časti. Výhoda tejto vety oproti vete 3.1.1 spočíva v tom, že počet koeficientov polynómu, ktoré treba vyrátať, je polynomiálny vzhľadom na dĺžku vstupu.

Veta 3.2.1. *Nech $n \geq 2$ je prirodzené číslo, p a r sú prvočísla, p delí n a $\ell = \lfloor 2\sqrt{r} \lg n \rfloor + 1$. Multiplikatívny rád d čísla n modulo r označme d. Nech platia nasledujúce predpoklady:*

(i) $(r, n) = 1$,

(ii) $p > r$,

(iii) $d > 4 \lg^2 n$,

(iv) pre každé a také, že $1 \leq a \leq \ell$ platí v $\mathbb{Z}_n[x]$

$$(x - a)^n = (x^n - a) \mod (x^r - 1).$$

Potom n je mocnina prvočísla p.

Dôkaz tejto vety bude rozdelený do viacerých liem. Vo zvyšku tejto kapitoly budeme stále predpokladať, že sú splnené podmienky (i) až (iv) z vety 3.2.1.

Budeme hovoriť, že polynóm $f(x) \in \mathbb{Z}_p[x]$ má vlastnosť (V) vzhľadom k prirodzenému číslu m, ak platí

$$[f(x)]^m = f(x^m) \mod (x^r - 1) \tag{V}$$

v $\mathbb{Z}_p[x]$.

Predpoklad (iv) nám teda vlastne hovorí, že polynómy $x - a$, kde $a = 1, \ldots, \ell$, majú vlastnosť (V) vzhľadom k číslu n. (Pretože $p \mid n$, ak uvedené identity platia v $\mathbb{Z}_n[x]$, tak platia aj v $\mathbb{Z}_p[x]$. Nemusí byť na prvý pohľad jasné, že všetky polynómy vystupujúce v (iv) sú prvkami $\mathbb{Z}_p[x]$. Stačí si však uvedomiť, že platia nerovnosti $r \geq d > 4 \lg^2 n$, $\sqrt{r} > 2 \lg n$, $r = \sqrt{r}\sqrt{r} > 2\sqrt{r} \lg n$. Z tejto nerovnosti dostaneme - vďaka tomu, že r je celé číslo - $p > r \geq \lfloor 2\sqrt{r} \lg n \rfloor + 1 = \ell$.) Pretože p je prvočíslo, tak podľa vety 3.1.1 má ľubovoľný polynóm prvého stupňa zo $\mathbb{Z}_p[x]$ vlastnosť (V) vzhľadom k číslu p.

Lema 3.2.2. *Ak polynóm $f(x)$ má vlastnosť* (V) *súčasne vzhľadom k m aj vzhľadom k m', tak má vlastnosť* (V) *aj vzhľadom k číslu mm'.*

Dôkaz. Predpokladajme, že v $\mathbb{Z}_p[x]$ platia rovnosti

$$[f(x)]^m = f(x^m) \mod (x^r - 1), \tag{3.1}$$

$$[f(x)]^{m'} = f(x^{m'}) \mod (x^r - 1). \tag{3.2}$$

Ak do (3.2) dosadíme x^m za x, dostaneme

$$[f(x^m)]^{m'} = f(x^{mm'}) \mod (x^{mr} - 1), \tag{3.3}$$

$$[f(x^m)]^{m'} = f(x^{mm'}) \mod (x^r - 1). \tag{3.4}$$

Z rovností (3.1) a (3.4) už vyplýva platnosť lemy. □

Lema 3.2.3. *Ak polynómy $f(x)$ a $g(x)$ majú vlastnosť* (V) *vzhľadom k číslu m, tak aj polynóm $f(x).g(x)$ má vlastnosť* (V) *vzhľadom k m.*

Dôkaz. Jednoduchým výpočtom overíme, že v $\mathbb{Z}_p[x]$ platí

$$[f(x)g(x)]^m = [f(x)]^m[g(x)]^m = f(x^m)g(x^m) \mod (x^r - 1).$$

□

Dôsledok 3.2.4. *Ak $f(x) \in \mathbb{Z}_p[x]$ je ľubovoľný polynóm tvaru $f(x) = \prod_{a=1}^{\ell}(x-a)^{\alpha_a}$, kde $\alpha_a \geq 0$, $a = 0, \ldots, \ell$, sú prirodzené čísla, tak $f(x)$ má vlastnosť* (V) *vzhľadom ku každému číslu tvaru $m = n^i p^j$, kde i, j sú prirodzené čísla.*

Lema 3.2.5. *Ak n nie je mocnina prvočísla p a prirodzené číslo w možno zapísať v tvare $n^i p^j$, tak sú čísla i a j jednoznačne určené. T.j. z rovnosti $n^i p^j = n^{i'} p^{j'}$ vyplýva $i = i'$ a $j = j'$.*

Dôkaz. Nech k je najväčšie prirodzené číslo, pre ktoré $p^k \mid n$ (t.j. $p^{k+1} \nmid n$). Potom $n = \alpha p^k$ pre nejaké prirodzené číslo $\alpha \neq 1$, pričom platí $(\alpha, p) = 1$. Z rovnosti $n^i p^j = n^{i'} p^{j'}$ dostaneme

$$\alpha^i p^{ik+j} = \alpha^{i'} p^{i'k+j'}.$$

Pretože $(\alpha, p) = 1$, z vyplýva z toho, že $i = i'$. Potom aj $p^{ik+j} = p^{i'k+j'}$, z čoho dostaneme $ik + j = i'k + j'$ a $j = j'$. □

Ako t budeme v ďalšom označovať počet prvkov množiny

$$I = \{n^i p^j \bmod r; i, j \in \mathbb{N}\}.$$

Pretože I obsahuje všetky prvky tvaru $n^i \bmod r$, platí $t \geq d$. (Číslo t je väčšie než multiplikatívny rád prvku n modulo r.) Samozrejme, keďže I obsahuje len zvyšky po delení číslom r, musí súčasne platiť $t \leq r$. Dokonca platí $t < r$, pretože $(n, r) = (p, r) = 1$ a r je prvočíslo, čiže I neobsahuje nulu.

Lema 3.2.6. *Položme*

$$E := \{n^i p^j : 0 \leq i, j \leq \lfloor \sqrt{t} \rfloor\}.$$

Ak n nie je mocnina prvočísla, tak množina E obsahuje viac než t prvkov. Špeciálne, E obsahuje dve čísla s rovnakým zvyškom po delení r.

Dôkaz. Zrejme platí $1 + \lfloor\sqrt{t}\rfloor > \sqrt{t}$, umocnením tejto nerovnosti dostaneme $(1 + \lfloor\sqrt{t}\rfloor)^2 > t$. Podľa lemy 3.2.5 pre každú dvojicu čísel $i, j \in \{0, 1, \ldots, \lfloor\sqrt{t}\rfloor\}$ dostaneme dva rôzne prvky množiny E. Teda máme $|E| \geq (1 + \lfloor\sqrt{t}\rfloor)^2 > t$. □

Dôkaz vety 3.2.1. Ak n je mocnina prvočísla, tak veta 3.2.1 platí. Predpokladajme teda, že n nie je mocnina prvočísla. Potom podľa lemy 3.2.6 existujú čísla i, j, i', j' také, že $0 \leq i, i', j, j' \leq \lfloor\sqrt{t}\rfloor$, $i \neq i'$ alebo $j \neq j'$ a platí

$$n^i p^j \equiv n^{i'} p^{j'} \pmod{r}.$$

Nech $m = n^i p^j$ a $m' = n^{i'} p^{j'}$. Bez ujmy na všeobecnosti môžeme predpokladať, že $m' \geq m$. Nech k je nezáporné celé číslo také, že $m' = m + kr$. Podľa dôsledku 3.2.4 máme

$$(x - a)^{m'} = (x^{m'} - a) = (x^{m+kr} - a) \mod (x^r - 1)$$

v $\mathbb{Z}_p[x]$ pre všetky $a = 1, \ldots, \ell$. Pretože $x^r = 1 \mod (x^r - 1)$ v $\mathbb{Z}_p[x]$, platí v $\mathbb{Z}_p[x]$ aj

$$(x - a)^{m'} = (x^m - a) \mod (x^r - 1).$$

Ďalej podľa dôsledku 3.2.4

$$(x^m - a) = (x - a)^m \mod (x^r - 1).$$

Teda v $\mathbb{Z}_p[x]$ platí

$$(x - a)^{m'} = (x - a)^m \mod (x^r - 1)$$

pre každé $a = 1, \ldots, \ell$.

Ukážeme, že $m = m'$. Z toho podľa lemy 3.2.5 vyplýva, že $i = i'$ a $j = j'$, čo je spor.

Zostáva teda už len dokázať, že $m = m'$. Označme $k := \min\{t, \ell\}$. Nech $h(x)$ je ireducibilný polynóm v $\mathbb{Z}_p[x]$, ktorý delí $\frac{x^r-1}{x-1}$. V $\mathbb{Z}_p[x]$ platí

$$(x - a)^{m'} = (x - a)^m \mod h(x) \tag{3.5}$$

pre každé $a = 1, \ldots, k$. Definujme podmnožinu $\mathbb{Z}_p[x]/(h(x))$

$$S := \left\{ \prod_{a=1}^{k} (x - a)^{\alpha_a} : \alpha_a \in \{0, 1\} \text{ pre každé } a = 1, \ldots, k \text{ a } \sum_{a=1}^{k} \alpha_a < k \right\}.$$

Uvažujme polynóm $z^{m'} - z^m$ v okruhu $(\mathbb{Z}_p[x]/(h(x)))[z]$. (Prvky tohto okruhu sú polynómy v neurčitej z, ktorých koeficientami sú polynómy zo $\mathbb{Z}_p[x]/(h(x))$.) Podľa dôsledku 3.2.4 je každý prvok z S koreňom $G(z)$. Neskôr ukážeme, že S má aspoň $2^{k-1} - 1$ prvkov. Zrejme m aj m' je menšie nanajvýš rovné $n^{\lfloor\sqrt{t}\rfloor} p^{\lfloor\sqrt{t}\rfloor} \leq \frac{1}{4} n^{2\sqrt{t}}$. (Prvočíslo p je vlastným deliteľom n, preto $p \leq \frac{n}{2}$. Súčasne $t \geq d > 4 \lg^2 n \geq 4$, a teda $\lfloor\sqrt{t}\rfloor \geq 2$.) Potom platí

$$\frac{1}{4} n^{2\sqrt{t}} = \frac{1}{4} 2^{2\sqrt{t} \lg n} \leq 2^{\lceil 2\sqrt{r} \lg n - 2 \rceil} \leq 2^{\lfloor 2\sqrt{r} \lg n \rfloor - 1} = 2^{\ell - 2} < 2^{\ell - 1} - 1.$$

Súčasne platí $t \geq d > 4\lg^2 n$, $\sqrt{t} > 2\lg n$, $t = \sqrt{t}\sqrt{t} > 2\sqrt{t}\lg n$. Keď umocníme 2 na ľavú a pravú stranu predchádzajúcej nerovnosti, získame odhady

$$n^{2\sqrt{t}} < 2^t, \ \frac{1}{4}n^{2\sqrt{t}} < 2^{t-2} < 2^{t-1} - 1.$$

Celkove teda dostávame

$$m, m' \leq n^{\lfloor\sqrt{t}\rfloor}p^{\lfloor\sqrt{t}\rfloor} < \frac{1}{4}n^{2\sqrt{t}} < 2^{\min\{t-1,\ell-1\}} - 1 = 2^{k-1} - 1.$$

Z toho vyplýva, že stupeň polynómu $G(z)$ je menší než $2^k - 1$. Teda ak $m \neq m'$, počet koreňov polynómu je väčší než jeho stupeň. Tento prípad podľa vety 2.2.2 nemôže nastať, lebo $\mathbb{Z}_p[x]/(h(x))$ je pole.

Treba teda ešte dokázať, že $|S| \geq 2^{k-1} - 1$. Platí $p > r \geq t$, a teda $p \geq k+1$. Z toho vyplýva, že polynómy $x - a$, $1 \leq a \leq k$ sú navzájom rôzne prvky $\mathbb{Z}_p[x]$ a každý z nich je zrejme ireducibilný. Pretože rozklad každého polynómu na ireducibilné polynómy v $\mathbb{Z}_p[x]$ je podľa vety 2.2.2 jednoznačný, aj všetky polynómy tvaru

$$\prod_{a=1}^{k}(x-a)^{\alpha_a},$$

kde $\alpha_a \in \{0, 1\}$ pre každé $a = 1, \ldots, k$ a $\sum_{a=1}^{k}\alpha_a < k$ sú navzájom rôzne polynómy v $\mathbb{Z}_p[x]$ a je ich $2^k - 1$. V prípade, že $h(x) = x - a$ pre niektoré a uvažujme len tie polynómy, ktoré neobsahujú člen $x - a$. Takýchto polynómov bude $2^{k-1} - 1$. Stačí ukázať, že tieto polynómy sú rôzne aj v $\mathbb{Z}_p[x]/(h(x))$.

Stupeň každého z týchto polynómov je najviac $k - 1$. Podľa dôsledku 3.2.4 každý polynóm z S má vlastnosť (V) vzhľadom ku každému číslu tvaru $s = n^i p^j$. Ak pre $g_1(x), g_2(x) \in S$ platí $g_1(x) = g_2(x)$ v $\mathbb{Z}_p[x]/(h(x))$, čiže

$$g_1(x) = g_2(x) \mod h(x),$$

tak aj $g_1^s(x) = g_2^s(x) \mod h(x)$ pre všetky s tvaru $n^i p^j$. Potom aj $g_1(x^s) = g_2(x^s) \mod h(x)$. Uvažujme polynóm $f(x) = g_1(x) - g_2(x)$ z $F := \mathbb{Z}_p[x]/(h(x))$. Prvky z F stotožňujeme s polynómami stupňa najviac t (reprezentantmi jednotlivých tried) a konštantné polynómy môžeme stotožniť s príslušnými konštantami. To znamená, že F je rozšírením poľa $\mathbb{Z}_p$, a teda f môžeme chápať ako polynóm nad poľom F.

Keďže podľa vety 2.2.6 rád x v multiplikatívnej grupe F^* poľa F sa rovná r, tak $x^s = x^{s'}$ platí v tomto poli práve vtedy, keď $s \equiv s' \pmod r$. Teda keď za s dosadzujeme postupne všetky čísla tvaru $n^i p^j$, dostaneme t rôznych prvkov $x^s \in F$. Pre každý z týchto prvkov platí $f(x^s) = g_1(x^s) - g_2(x^s) = 0$, čiže f je nenulový polynóm stupňa menšieho ako t, ktorý má t koreňov, čo je spor podľa vety 2.2.3. □

Kapitola 4

Popis AKS-algoritmu

V tejto kapitole popíšeme AKS-algoritmus. Ukážeme, že z vety 3.2.1 vyplýva jeho správnosť a odhadneme jeho časovú zložitosť.

4.1 Popis algoritmu

AKS-algoritmus, ktorý rozhodne, či dané číslo n je prvočíslo, môžeme neformálne popísať pomocou nasledujúcich krokov:

1. Ak $n = a^b$ pre nejaké prirodzené čísla $a, b > 1$, tak vrátime `FALSE`.

2. Hľadáme najmenšie prvočíslo r také, že platí buď $(n, r) > 1$ alebo $(n, r) = 1$ a multiplikatívny rád d čísla n modulo r je viac než $4 \lg^2 n$.

3. Ak $r = n$, tak vrátime `TRUE`.

4. Ak $(n, r) > 1$ tak vrátime `FALSE`.

5. Postupne pre a od 1 po $\ell = \lfloor 2\sqrt{r} \lg n \rfloor + 1$ testujeme rovnosť

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

   v $\mathbb{Z}_n[x]$. Akonáhle nájdeme najmenšie a, pre ktoré táto rovnosť neplatí, vrátime `FALSE`.

6. Ak rovnosť platí pre všetky $a = 1, \ldots, \ell$, tak vrátime `TRUE`.

V kroku 1 môžeme použiť funkciu `Is_Perfect_Power()`, ktorá je popísaná v leme 2.3.9.

V kroku 2 vlastne postupne prechádzame čísla menšie ako n, ale budeme testovať iba tie z nich, ktoré sú prvočísla. Na testovanie prvočíselnosti stačí použiť niektorý jednoduchý algoritmus - napríklad Eratostenovo sito popísané v leme 2.3.8. Odhad pre multiplikatívny rád čísla n modulo r overíme jednoducho umocnením.

Krok 5 je popísaný v leme 2.3.7.

Poznamenajme, že sa v literatúre vyskytli aj ďalšie varianty AKS-algoritmu. Napríklad v kroku 2 sa nehľadajú len prvočísla, ale ľubovoľné čísla r s uvedenými vlastnosťami. Ďalšia modifikácia bol algoritmus, v ktorom sa navyše kládli iné podmienky - dolný odhad na najväčší prvočíselný faktor q čísla $r-1$ a $n^{\frac{r-1}{q}} \not\equiv 1 \pmod{r}$.

Ďalej sa budeme venovať časovej zložitosti a správnosti uvedeného algoritmu. Pri zistení časovej zložitosti bude najdôležitejšie dokázať existenciu prvočísla r, ktoré má požadované vlastnosti a odhadnúť jeho veľkosť.

Existencia r sa bude využívať aj v dôkaze správnosti algoritmu. Na dôkaz správnosti vlastne len treba overiť, že prvočíslo r, ktoré nájdeme v algoritme a vhodný prvočiniteľ p čísla n spĺňajú predpoklady vety 3.2.1.

4.2 Časová zložitosť algoritmu

Lema 4.2.1. *Pre prirodzené čísla $n > 1$, $m \geq 1$ najmenšie prvočíslo r také, že $r \mid n$ alebo multiplikatívny rád d čísla n modulo r je väčší než m má veľkosť $O(m^2 \lg n)$.*

Dôkaz. Prvočíslo r budeme nazývať „dobrým“, ak spĺňa podmienky uvedené v leme, inak ho budeme volať „zlé“ prvočíslo. Ak r je zlé, tak buď $r \nmid n$ alebo $r \mid (n^d - 1)$ pre nejaké $1 \leq d \leq m$. Teda pre každé zlé prvočíslo platí

$$r \mid \prod_{d=1}^{m} (n^d - 1).$$

Ak všetky prvočísla až po číslo x sú zlé, potom aj súčin všetkých týchto prvočísel delí $\prod_{d=1}^{m} (n^d - 1)$ a dostávame

$$\prod_{p \leq x} p \leq \prod_{d=1}^{m} (n^d - 1) \leq \prod_{d=1}^{m} n^d = n^{\frac{m(m+1)}{2}}.$$

Zlogaritmovaním tejto nerovnosti dostaneme

$$\sum_{p \leq x} \ln p \leq \sum_{d=1}^{m} d \ln n = \frac{m(m+1)}{2} \ln n.$$

Ale podľa dôsledku 2.1.10 platí $\sum_{p \leq x} \ln p \geq Ax$ pre nejaké reálne číslo $A > 0$, z čoho dostaneme, že

$$x \leq \frac{\frac{m(m+1)}{2} \ln n}{A} = O(m^2 \lg n).$$

□

Zvlášť odvodíme časovú zložitosť pre najpodstatnejšie kroky algoritmu:

Lema 4.2.2. *Ak r je číslo, ktoré nájdeme v kroku 2 AKS-algoritmu, tak krok 2 trvá nanajvýš $\tilde{O}(r \lg^2 n)$.*

Dôkaz. Test v kroku 2 bude treba robiť iba pre r čísel. Overenie, či n a r sú nesúdeliteľné znamená overiť, či r delí n. Celočíselné delenie trvá $O^{\sim}(\lg n)$ (pozri lemu 2.3.2 a 2.3.1). Na overenie, či multiplikatívny rád čísla n modulo r je viac než $4\lg^2 n$ potrebujeme najviac $4\lg^2 n$ násobení v $\mathbb{Z}_r$. Výpočty v $\mathbb{Z}_r$ môžeme realizovať pomocou celočíselného násobenia, podľa liem 2.3.1 a 2.3.2 vrátene incializácie na hodnotu $n \mod r$ trvá tento krok najviac $O^{\sim}(\lg n + \lg^2 n \lg r)$ pri použití FFT, resp. $O(\lg n + \lg^2 n \lg^2 r)$ pri pomalšom násobení. Celkovo dostávame pre krok 2 zložitosť $O^{\sim}(r \lg^2 n)$. □

Lema 4.2.3. *Ak r je číslo, ktoré nájdeme v kroku 2 AKS-algoritmu, tak krok 5 trvá nanajvýš $O^{\sim}(r^{1.5} \lg^3 n)$ (FFT), resp. $O(r^{2.5} \lg^4 n)$ (pomalšie násobenie).*

Dôkaz. V kroku 5 potrebujeme overiť $\ell = \lfloor 2\sqrt{r} \lg n \rfloor + 1$ rovností. Výpočet koeficientov polynómu $(x-a)^n \mod (x^r - 1)$ v $\mathbb{Z}_n[x]$ trvá podľa lemy 2.3.7 $O^{\sim}(r \lg^2 n)$.

Na základe rovnosti $x^r = 1 \mod (x^r - 1)$ sa ľahko overí, že ak $n = ur + v$, $0 \leq v < r$, tak $x^n - a = (x^r)^u x^v - a = x^v - a \mod (x^r - 1)$, preto na výpočet ľavej strany rovnosti stačí vypočítať $v = n \mod r$, čo trvá podľa lemy 2.3.2 $O^{\sim}(\lg n)$.

Teda overenie každej rovnosti trvá $O^{\sim}(r \lg^2 n)$ a celková zložitosť kroku 5 je teda $O^{\sim}(r^{1.5} \lg^3 n)$.

Pri použití pomalšieho násobenia potrvá overenie jednej rovnosti $O(r^2 \lg^3 n)$, čo vedie k celkovej zložitosti $O(r^{2.5} \lg^4 n)$. □

Veta 4.2.4. *AKS-algoritmus má časovú zložitosť $O^{\sim}(\lg^{10.5} n)$ (FFT) resp. $O(\lg^{16.5} n)$ (pomalšie násobenie).*

Dôkaz. Krok 1 má podľa lemy 2.3.9 časovú zložitosť $O^{\sim}(\lg^3 n)$.

Podľa lemy 4.2.1 číslo r nájdené v kroku 2 bude mať veľkosť najviac $O(\lg^5 n)$. Krok 2 trvá podľa lemy 4.2.2 $O^{\sim}(r \lg^4 n) = O^{\sim}(\lg^9 n)$. Časová zložitosť je podmienená hlavne krokom 5, ktorý podľa lemy 4.2.3 trvá $O^{\sim}(r^{1.5} \lg^3 n) = O^{\sim}(\lg^{10.5} n)$.

V prípade pomalšieho násobenia dostaneme zložitosť $O(r^{2.5} \lg^4 n) = O(\lg^{16.5} n)$. □

4.3 Správnosť AKS-algoritmu

Veta 4.3.1. *AKS-algoritmus vráti hodnotu* `TRUE` *práve vtedy, keď jeho vstup n je prvočíslo.*

Dôkaz. Ak n je prvočíslo, tak algoritmus nemôže vrátiť `FALSE` v krokoch 1, 4 a podľa vety 3.1.1 ani v kroku 5. Teda vráti `TRUE` v kroku 3 alebo 6.

Predpokladajme, že n je zložené číslo. Ak n je netriviálnou mocninou iného prirodzeného čísla, tak algoritmus vráti `FALSE` v kroku 1. Ak nevráti `FALSE` v kroku 1 ani v kroku 4, tak sme našli v kroku 2 prvočíslo r také, že platí $(r, n) = 1$ a multiplikatívny rád d čísla n modulo r je viac než $4\lg^2 n$. Súčasne pre ľubovoľný prvočiniteľ p čísla n platí $p > r$, inak by sme vyskúšali prvočíslo p už v kroku 2 a vrátili `FALSE`. Potom AKS-algoritmus vráti `FALSE` v kroku 5. Ak by totiž platili všetky rovnosti, ktoré v tomto kroku testujeme, tak sú splnené všetky predpoklady vety 3.2.1 a z nej dostaneme, že n je mocnina prvočísla. To je však spor, lebo takýto prípad sme vylúčili už v kroku 1. □

4.4 Vylepšenie odhadu na časovú zložitosť

Zatiaľ sme používali len pomerne elementárne výsledky teórie čísel. Použitím odhadov dokázaných v analytickej teórii čísel možno vylepšiť odhad na prvočíslo r, ktoré získame v AKS-algoritme. Ide konkrétne o nasledujúcu vetu, ktorú možno nájsť napríklad v článkoch [7] a [15].

Veta 4.4.1. *Označme ako $\pi'(n)$ počet prvočísel p takých, že $p \leq x$ a $p-1$ má prvočiniteľ väčší alebo rovný $p^{\frac{2}{3}}$. Potom existuje konštanta $c > 0$ taká, že*

$$\pi'(x) \geq \frac{cx}{\lg x}$$

pre všetky dostatočne veľké reálne čísla x.

Pomocou tejto vety možno získať odhad $r = O(\lg^3 n)$, z ktorého potom vyplýva:

Veta 4.4.2. *Časová zložitosť AKS-algoritmu je $O\tilde{}(\lg^{7.5} n)$.*

Poznamenajme, že sú známe aj ďalšie vylepšenia odhadov na časovú zložitosť. Napríklad v článku [11] je uvedená modifikácia AKS-algoritmu, ktorá pre pomerne veľkú triedu čísel má časovú zložitosť $O\tilde{}(\lg^4 n)$.

Viaceré vylepšenia časovej zložitosti sa opierajú o niektorú teoreticko-číselnú hypotézu, na základe ktorej sa dá vylepšiť odhad na číslo r, ktoré nájdeme v kroku 2 AKS-algoritmu.

Hypotéza 4.4.3 (Artin). *Pre každé číslo n, ktoré nie je druhou mocninou prirodzeného čísla, nech $B(m)$ je počet prvočísel $r \leq m$ takých, že multiplikatívny rád d čísla n modulo r je $r-1$. Potom platí $B(m) \sim A(n)\frac{m}{\ln m}$ a $A(n) > 0.35$.*

Je známe, že Artinova hypotéza platí za predpokladu zovšeobecnenej Riemannovej hypotézy.

Hypotéza 4.4.4 (Sophie-Germain). *Ak $S(m)$ je počet prvočísel $r \leq m$ takých, že aj $2r+1$ je prvočíslo, tak $S(m) \sim \frac{2C_2 m}{\ln^2 m}$, C_2 je približne 0.66.*

Ak je platí niektorá z predchádzajúcich 2 hypotéz, tak potom možno ukázať, že $r = O(\lg^2 n)$, čo vedie k časovej zložitosti $O\tilde{}(\lg^6 n)$.

Nasledujúca hypotéza je uvedená v [12] a je tam overená pre všetky $r \leq 100$ a $n \leq 10^{10}$.

Hypotéza 4.4.5 (Bhattacharjee, Prandey). *Ak r je prvočíslo, $r \nmid n$ a v $\mathbb{Z}_n[x]$ platí $(x-1)^n = x^n - 1 \mod (x^r - 1)$, tak buď n je prvočíslo alebo $n^2 \equiv 1 \pmod r$.*

Ak platí táto hypotéza, tak možno veľmi jednoducho implementovať algoritmus na testovanie prvočíselnosti tak, že beží dokonca v čase $O\tilde{}(\lg^3 n)$.

Lema 4.4.6. *Ak platí hypotéza 4.4.5, tak algoritmus* `Primes_BP` *zistí či n je prvočíslo v čase $O\tilde{}(\lg^3 n)$ (pri použití FFT) resp. $O(\lg^5 n)$ (pomocou pomalšieho násobenia).*

```
bool Primes_BP(int n) {
  int r,k;

    r=2;
  do {
    r++;
    k=n%r;
    if (k==0) {//r | n
      if (r==n)
        return(true)
      else
        return(false);
    }
  } while ((k*k)%r==1);


  if ((x-1)^n = x^n - 1 mod (x^r - 1) v Z_n[x])
    return(true);
  else
    return(false);
}
```

4.5 Implementácia AKS-algoritmu

Na implementáciu sme použili knižnicu NTL [26], ktorej autorom je V. Shoup. Táto knižnica obsahuje triedy určené na prácu s veľkými číslami a polynómami a na počítanie v $\mathbb{Z}_p$, $\mathbb{Z}_p[x]$ a $\mathbb{Z}_p[x]/(h(x))$, kde $h(x)$ je ľubovoľný pevne zvolený polynóm. To znamená, že pomocou tejto knižnice sme mohli jednoducho naprogramovať krok 5 AKS-algoritmu. Pretože polynóm, s ktorým pracujeme, má špeciálny tvar $x^r - 1$, z hľadiska efektivity programu bolo výhodnejšie vytvoriť vlastné funkcie na počítanie modulo $x^r - 1$ než používať funkcie z knižnice NTL, ktoré sú určené pre výpočty s ľubovoľným polynómom.

Test v kroku 1 sme naprogramovali tak, ako je popísaný v leme 2.3.9. V kroku 2 sme použili funkciu `Is_Prime` z lemy 2.3.8. Samozrejme namiesto `int` sme použili objekty triedy `ZZ`, ktorá je v knižnici NTL použitá na reprezentáciu veľkých čísel.

Knižnica NTL má obmedzenie, že stupeň polynómu je typu `long`, t.j. najviac $2^{63}-1$. Teda túto knižnicu môžeme použiť na test v kroku 5 len ak r nepresahuje túto hodnotu. Z praktického hľadiska to však nie je obmedzujúce, pretože $r \leq \lceil 16 \lg^5 n \rceil$ (pozri [2, Lemma 4.3]). Potom máme pre $\lg n < 2^{11}$ nerovnosti $\lceil \lg^5 n \rceil < 2^{59}$, $r \leq \lceil 16 \lg^5(n) \rceil < 2^{63}$, čiže môžeme bezpečne počítať pre čísla, ktoré majú menej ako 2048 číslic v binárnej sústave. Za predpokladu platnosti hypotézy 4.4.4 platí dokonca $r \leq \lfloor 4 \lg^2 n \rfloor$, čiže ak platí táto hypotéza, môžeme test používať pre čísla do s $\lfloor \lg n \rfloor \leq 2^{30}$, čiže s 2^{30} binárnymi ciframi.

4.6 Testovanie implementácie

Pri testovaní našej implementácie AKS-algoritmu sme zistili, že test beží oveľa rýchlejšie pre zložené čísla ako pre prvočísla. Časovo najnáročnejším krokom je overovanie kongruencií - teda čas behu algoritmu závisí od veľkosti čísla r, čiže nerastie úplne striktne, ale niekedy aj pre veľké n môžeme mať „šťastie“ a algoritmus zbehne pomerne rýchlo.

Testy sme spúšťali na počítači Pentium4 s frekvenciou 2GHz, 512 MB RAM. Nasledujúca tabuľka zobrazuje časy dosiahnuté pre vstupy tvaru $2^n + k$, kde n je nepárne číslo a $2^n + k$ je najbližšie prvočíslo, ktoré nasleduje po tomto čísle. Porovnávali sme efektivitu našej implementácie z imlementáciou z [16] využívajúcou knižnicu NTL a z implementáciou z [6] v C++. (Na [16] možno tiež nájsť implementácie využívajúce iné knižnice a v práci [6] bola vytvorená aj implementácia v Jave.) Tieto implementácie pracujú so vstupom ako s celým číslom typu `long`, preto sme mohli testovať len obmedzený počet hodnôt. Obe implementácie sme upravili tak, aby formát vstupu a výstupu rovnaký ako pri našej implementácii. Všetky implementácie dosahovali výrazne horšie časy než najjednoduchší algoritmus `Is_Prime`, čo svedčí o tom, že táto podoba algoritmu nie je vhodná pre praktické účely. Čas uvádzame vo formáte hodiny:minúty:sekundy.

$n = \lfloor \lg p \rfloor$	`Primes`	Gallot	Aoyama	`Is_Prime`
3	0	0	0	0
5	0	0	0	0
7	0	0	0	0
9	2	5	1	0
11	8	11	17	0
13	27	44	5:22	0
15	1:01	1:20	30:40	0
17	3:52	6:17	-	0
19	6:20	9:21	-	0
21	10:16	13:48	-	0
23	26:02	37:45	-	0.01
25	36:29	51:22	-	0.01
27	49:20	1:06:37	-	0.01
29	1:06:13	1:26:04	-	0.01
31	1:55:06	2:28:34	-	0.03
33	3:23:00	-	-	0.07
35	5:28:51	-	-	0.13

Algoritmus `Primes_BP` založený na hypotéze 4.4.5 dosahoval veľmi rýchle časy. V nasledujúcej tabuľke je jeho porovnanie s výsledkami Rabin-Millerovho testu pre Mersennove prvočísla $p = 2^n - 1$ s exponentom pod 10000. (Použili sme implementáciu tohto testu, ktorá je súčasťou knižnice NTL.)

Na základe tejto tabuľky možno usúdiť, že ak by sa podarilo dokázať hypotézu 4.4.5 alebo nejakým iným spôsobom výrazne znížiť počet kongruencií, ktoré sa overujú pri testovaní prvočíselnosti, dostali by sme pomerne efektívny algoritmus, stále však dosahuje výrazne horšie výsledky než Rabin-Millerov test.

n	`Primes_BP`	Rabin-Miller
3	0	0
5	0	0
7	0	0
13	0	0
19	0	0
31	0	0
61	0	0
89	0	0
107	0	0
127	0	0
521	0	0
607	0	0
1279	2	1
2203	10	6
2281	1:06	7
3217	1:21	17
4253	-	38
4423	-	42

Literatúra

[1] L. M. Adleman, C. Pomerance, R. S. Rumely: *On distinguishing prime numbers from composite numbers,* Ann. Math. **117** (1983), 173–206.

[2] M. Agrawal, N. Kayal, N. Saxena: *PRIMES is in P,* `http://www.cse.iitak.ac.in/users/manindra/index.html`, 2002.

[3] A. O. L. Atkin: *Lecture notes of a conference,* Boulder Colorado (1986).

[4] A. O. L. Atkin, F. Morain: *Elliptic curves and primality proving,* Math. Comp. **61** (1993), 29–68.

[5] W. R. Alford, A. Granville, C. Pomerance: *There are infinitely many Carmichael numbers,* Ann. Math. **139** (1994), 703–722.

[6] T. Aoyama: *Polynomial Time Primality Testing Algorithm,* Master's Project, 2003, `http://www.cs.rit.edu/~axt/ms_project/ms_project.html`.

[7] R. C. Baker, G. Harman: *The Brun-Titchmarsch Theorem on average,* Proceedings of a conference in Honor of Heini Halberstam, Volume 1 (1996), 39–107.

[8] D. J. Bernstein: *Detecting perfect powers in essentially linear time,* Mathematics of Computation **67** (1998), 1253–1283.

[9] D. J. Bernstein: *Fast multiplication and its applications,* `http://cr.yp.to/papers.html#multapps`.

[10] D. J. Bernstein: *Proving primality in essentially quartic expected time,* `http://cr.yp.to/papers.html#quartic`.

[11] P. Berrizbeitia: *Sharpening "Primes is in P" for a large family of numbers,* `http://arxiv.org/abs/math.NT/0211334`

[12] R. Bhattacharjee, P. Pandey: *Primality Testing. Techical Report,* IIT Kanpur, 2001. `http://www.cse.iitk.ac.in/research/btp2001/primality.html`

[13] H. Cohen, A. K. Lenstra: *Primality testing and Jacobi sums,* Math. Comp. **42** (1984), 297–330.

[14] R. Crandall, C. Pomerance: *Prime Numbers, a Computational Perspective,* Springer-Verlag, New York, 2001.

[15] E. Fouvry: *Théorème du Brun-Titchmarsch; aplication au théorème de Fermat,* Invent. Math., **79** (1985), 383–407.

[16] Y. Gallot: Implementation of the AKS algorithm, `http://perso.wanadoo.fr/yves.gallot/src`.

[17] S. Goldwasser, J. Kilian: *Almost all primes can be quickly certified,* STOC86, Proceedings of the 18th Annual ACM Symposium on the Theory of Computing (Berkeley, 1986), ACM, 316–629, New York, 1986.

[18] G. H. Hardy, E. M. Wright: *Introduction to the Theory of Numbers,* Oxford, 1954.

[19] W. W. L. Chen: *Elementary Number Theory,* Lecture notes, `http://www.maths.mq.edu.au/~wchen`.

[20] Q. Cheng: *Primality proving via one round in ECPP and one iteration in AKS,* Crypto 2003, Santa Barbara, LNCS 2729.

[21] T. Katriňák, M. Gavalec, M. Gedeonová, J. Smítal: *Algebra a teoretická aritmetika 1,* UK, Bratislava, 2002.

[22] D.E. Knuth: *The Art of Computer Programming, volume 2: Seminumerical algorithms,* Adison Wesley, Massachusetts, 1998.

[23] M. Kolibiar a kol.: *Algebra a príbuzné disciplíny*, Alfa, Bratislava, 1992.

[24] D. N. Lehmer: *An extended theory of Lucas' functions,* Ann. Math. **31** (1930), 419–448.

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

[26] NTL: A library for doing Number Theory, `http://www.shoup.net/ntl`.

[27] M. Křížek, L. Somer: *Preudoprvočísla,* Pokroky matematiky, fyziky a astronómie **48(2)** (2003), 143–151.

[28] M. O. Rabin: *Probabilistic agorithm for testing primality,* J. Number Theory **12** (1980), 128–138.

[29] M. Nair: *On Chebyshev-type inequalities for primes,* Amer. Math. Monthly **89** (1982), 126–129.

[30] V. Shoup: *A Computational Introduction to Number Theory and Algebra (beta version 2),* `http://shoup.net/ntb`.

[31] M. Smid: *Primality testing in polynomial time,* `http://www.scs.carleton.ca/~michiel/primes.ps.gz`, 2003.

[32] A. Valachová-Rusnáková: *Prvočísla,* diplomová práca, MFF UK, Bratislava, 1981.

[33] F. P. Voloch: *On some subgroups of the multiplicative groups of finite rings,* J. de Théorie des Nombres de Bordeaux, to appear.