POLLARDOVA ρ METODA

za praštevilski razcep naravnih števil

Projekt pri predmetu Kriptografija in teorija kodiranja II

Avtor: Martin Ambrožič
Mentor: Aleksandar Jurišić

Problem praštevilskega razcepa naravnega števila

Zanima nas naslednja naloga: Zapisati dano naravno število n kot produkt praštevil ali njihovih potenc. Sila enostavna formulacija, ki je ravno zaradi svoje preprostosti pritegnila zanimanje mnogih matematikov že vse od Antike naprej. Izkaže se namreč, da je to zelo težak problem, za katerega ne poznamo učinkovitega algoritma.

Težavnost določenih številskih problemov je pogosto osnova za algoritme v kriptografiji. Težavnost problema praštevilskega razcepa se izkorišča na primer pri kriptosistemu RSA. Kriptogram x iz besedila m namreč izračunamo kot $x \equiv m^e (mod\, n)$, kjer je e javni kodirni ključ, modul n pa je produkt dveh praštevil: $n = pq$. Obratno izračunamo iz kriptograma x izvorno besedilo m kot $m \equiv x^d (mod\, n)$, kjer je d tajni dekodirni ključ, znan samo lastniku.
Za par ključev (e, d) mora zato veljati $ed \equiv 1\big(mod\,(p-1)(q-1)\big)$. Praštevilski razcep modula n ne sme biti znan, saj je sicer moč iz javnega ključa e učinkovito izračunati pripadajoči zasebni ključ d. Če torej predpostavimo, da je problem praštevilskega razcepa števila n dovolj težak, nam je zagotovljena določena stopnja varnosti. Problem praštevilskega razcepa torej ne samo zanimiva in elegantno zastavljena matematična naloga, temveč je podlaga za zelo pomemben in razširjen kriptosistem in ima zato njegov študij še toliko večjo vrednost.

Poglejmo si nekaj osnov.

Zapis, $n = q_1^{\alpha_1} q_2^{\alpha_2} \dots q_k^{\alpha_k}$, kjer so q_1, q_2, …, q_k praštevila in $\alpha_i > 0$ za vsak $i = 1, 2, \dots, k$, imenujemo praštevilski razcep števila n, in je enoličen, če nas ne zanima vrstni red faktorjev. Če želimo poiskati praštevilski razcep za dani n, je potrebno poiskati le enega od praštevilskih faktorjev, denimo q, saj ostale lahko izračunamo rekurzivno kot praštevilske faktorje števila $\frac{n}{q}$.

Najpreprostejša metoda, s katero si lahko pomagamo pri določanju praštevilskega razcepa, je poskusno deljenje. Naše število n poskušamo deliti z vsemi praštevili med 1 in $\sqrt{n}$. Če se katero od deljenj izide, smo našli praštevilski faktor, sicer je n praštevilo. S praštevili, večjimi od $\sqrt{n}$, ni treba poskušati. Če ima n prafaktor $q > \sqrt{n}$, pri čemer $q \neq n$, je tedaj $1 < \frac{n}{q} < \sqrt{n}$, kar pomeni, da je bodisi $\frac{n}{q}$ praštevilski delitelj za n, manjši od $\sqrt{n}$, bodisi je $\frac{n}{q}$ produkt takih praštevilskih deliteljev.

Poskusno deljenje je učinkovito za majhna števila, vendar pa je pri velikih številih prepočasno. Njegova časovna zahtevnost je namreč $O(f(\log n)^2)$, kjer je f najdeni praštevilski faktor.

Zato si bomo ogledali učinkovitejšo Pollardovo ρ metodo za praštevilski razcep, ki ima pričakovano časovno zahtevnost $\mathcal{O}(\sqrt{f}(\log n)^2)$. Ta metoda ni med najhitrejšimi danes znanimi metodami, je pa lahko znatno hitrejša od poskusnega deljenja in je razmeroma enostavno razumljiva.

Metoda

Ideja Pollardove ρ metode je sledeča:
Naj bo $N \epsilon \mathbb{N}$ sestavljeno naravno število in naj bo $p(x)$ polinom s celimi koeficienti. Definiramo zaporedje z začetno vrednostjo x_0 in rekurzivno formulo $x_n = p(x_{n-1}) \ (mod\, N)$.
Različnih ostankov po modulu N je natanko N, zato je neizogibno zaporedje $(x_n)_n$ periodično. Označimo njegovo periodo s T, tj.:
$x_{i+T} = x_i$

za vsak $i > U$, kjer je U predperioda, torej število členov zaporedja, preden začne veljati periodičnost.

Primer: zaporedje 1, 2, 3, 4, 3, 4, 3, 4, 3, 4, ... ima periodo $T = 2$ in predperiodo $U = 2$.

Naj bo zdaj q nek neznani praštevilski delitelj števila N. Definiramo novo zaporedje $(y_n)_n$ s členi zaporedja $(x_n)_n$ po modulu q, torej $y_n = x_n \ (mod\, q)$. Velja $x_n \equiv p(x_{n-1}) \ (mod\, N)$
Ker pa q deli N, tudi: $x_n \equiv p(x_{n-1}) \ (mod\, q)$
in po definiciji $(y_n)_n$ $y_n \equiv p(y_{n-1}) \ (mod\, q)$

Zaradi končnosti zaloge vrednosti je zaporedje $(y_n)_n$ tudi periodično in ima periodo največ q, kar je gotovo manj od N. To lastnost omenjenih zaporedij izkoristimo za iskanje deliteljev števila N, namreč:
naj bo t perioda zaporedja $(y_n)_n$. Za dovolj velike indekse i torej velja:

$y_{i+t} \equiv y_i (mod\, q)$ natanko tedaj pa tudi: $x_{i+t} \equiv x_i (mod\, q)$

zaradi definicije zaporedja $(y_n)_n$. To pa po definiciji kongurence pomeni:

$x_{i+t} - x_i = kq$
za nek $k \in \mathbb{Z}$. Ker je q pričakovano precej manjši od N, lahko pričakujemo, da razlika $x_{i+t} - x_i$ ne bo deljiva z N, in tedaj velja, da je $GCD(x_{i+t} - x_i, N)$ netrivialen delitelj števila N.
Na podlagi te izpeljave že lahko zapišemo osnovno različico algoritma.

Pollardova ρ metoda – osnovna različica

```
k := 0          (indeks zaporedja x_k)
x_0 := x_0      (prvi člen izberemo sami)

ponavljaj
     k := k + 1
     x_k := p(x_k-1) mod N

     za 0 ≤ j ≤ k-1:
          g := GCD(x_k-x_j, N)
          če je 1 < g < N, vrni g in končaj
```

Algoritem je verjetnostna metoda, kar pomeni, da ni zagotovila ustavljanja. Poleg tega preverjamo največji skupni delitelj za vse razlike od 0 do $k-1$ za vsak k, kar pomeni kvadratno število operacij, kar pa ni zaželjeno. Izkaže se namreč, da je moč najti cikel v zaporedju $(x_n)_n$ tudi z manjšim številom zahtevanih operacij.

Uvedemo še eno zaporedje $(z_n)_n$ z začetno vrednostjo x_0 in rekurzivno definiramo $z_n = p\big(p(z_{n-1})\big)\ (mod\ N)$.
Tedaj velja:
$z_k = x_{2k}$
in če je k dovolj velik večkratnik periode t:
$z_k = x_{2k} = x_k$

in je zato $GCD(z_k - x_k, N)$ iskani delitelj.

Zapišimo to v izboljšani različici algoritma.

Pollardova ρ metoda – izboljšana različica

```
k := 0
x_0 := x_0
z_0 := x_0

ponavljaj
     k := k + 1
     x_k := p(x_k-1) mod N
     z_k := p(p(z_k-1)) mod
     g := GCD(x_k-x_j, N)
     če je 1 < g < N, vrni g in končaj
```

Slaba stran tega algoritma je ta, da potrebujemo v vsaki iteraciji tri izračune polinoma p, ki je načeloma lahko katerikoli polinom s celimi koeficienti.

Nadaljnjo izboljšavo algoritma je predlagal Brent in sicer je pokazal, da je moč detekcijo cikla v zaporedju $(x_n)_n$ izvesti tako, da trenutno vrednost x_n primerjamo z zadnjo vrednostjo v zaporedju, katere indeks je potenca števila 2. Brent je poizkušal

tudi z drugimi bazami, a je ugotovil, da je baza 2 blizu optimalne. Dokaz, da ta metoda deluje, je moč najti v [1].

Zapišimo še ta algoritem.

Pollardova ρ metoda – Brentova izboljšava

```
k := 0
x₀ := x₀
u := 0          (zadnja potenca št. 2)

ponavljaj
     če je k potenca št. 2
          u := k
     k := k + 1
     xₖ := p(xₖ₋₁) mod N
     g := GCD(xₖ-xᵤ, N)
     če je 1 < g < N, vrni g in končaj
```

Vidno je, da smo se rešili dveh izračunov funkcijskih vrednosti, saj zdaj računamo samo še z originalnim zaporedjem $(x_n)_n$, poleg tega pa tudi ni več kvadratnega števila primerjav njegovih členov, ki je bilo problematično v prvem algoritmu. Tu sicer nimamo več zagotovila, da bomo cikel odkrili pri najmanjših mogočih indeksih zaporedja $(x_n)_n$, saj ne primerjamo vseh členov med seboj, vendar pa se vseeno izkaže, da je ob prepodstavki, da je generirano zaporedje $(x_n)_n$ psevdonaključno, potrebno $\mathcal{O}(\sqrt{f}(\log n)^2)$ operacij, da dosežemo verjetnost uspeha $\frac{1}{2}$. Če upoštevamo še, da je najdeni faktor f velikosti največ $\sqrt{n}$, lahko število potrebnih operacij zapišemo kot $\mathcal{O}\left(n^{1/4}(\log n)^2\right)$

Možne so še dodatne izboljšave. Računanje največjega skupnega delitelja števil je razmeroma zahtevna operacija, zato lahko prihranimo na času tako, da ne računamo $GCD(x_k - x_u, N)$ za vsako razliko $x_k - x_u$, ampak računamo produkte (denimo) desetih zaporednih razlik in izračunamo $GCD(\prod(x_k - x_u), N)$. Če je ta enak 1, so tudi vsi $GCD(x_k - x_u, N)$ enaki 1. Če pa je večji od 1, pa je potrebno iti nazaj in preveriti, če smo našli kak netrivialni delitelj za N. Uspeh namreč tudi tedaj še ni zagotovljen, saj se lahko zgodi tudi, da je $GCD(x_k - x_u, N) = N$.

Polinom p(x)

O polinomu, ki generira zaporedje v Pollardovi metodi nismo veliko govorili, čeprav lahko bistveno vpliva na učinkovitost algoritma. Poglejmo, kaj je moč povedati o njegovem izboru.

Pollardova ρ metoda se zanaša na pojav cikla v zaporedju $(x_n)_n$, ki smo ga definirali zgoraj. Njena časovna zahtevnost je odvisna od dolžine tega cikla skupaj z repom, tj. elementov zaporedja preden se začne periodičnost. Dolžina cikla v generiranem zaporedju pa je povezana (preko zaporedja $(y_n)_n$, ki smo ga definirali v uvodu) z dolžinami ciklov generiranega zaporedja po praštevilskem modulu, v resnici po modulu nekega praštevilskega delitelja števila N.

Izbrati želimo torej polinom, ki ima v povprečju karseda kratke dolžine ciklov, pri čemer predpostavljamo, da ne vemo ničesar o številu, ki ga želimo faktorizirati. V nasprotnem primeru, če imamo neko znanje o deliteljih števila, ga lahko uporavimo pri izboru polinoma. V skrajni situaciji, ko poznamo kakega delitelja q števila N, lahko s polinomom $p(x) = qx$ praktično takoj dobimo rezultat, saj velja:

$x_k = qx_{k-1} = q^2 x_{k-2}$
in zato:
$x_k - x_{k-1} = q(x_{k-1} - qx_{k-2}) \equiv 0 \ (mod\ q)$
oziroma:
$x_k \equiv x_{k-1} \ (mod\ q)$

kar pa smo na začetku želeli (gre kar za zaporedje $(y_n)_n$, zaradi kongruentnosti po modulu delitelja).

Ampak ta primer je v resnici absurden, saj če poznamo enega od deliteljev, ga seveda ne bomo ponovno iskali s Pollardovo ρ metodo. Gre samo za prikaz izkoriščanja znanja o deliteljih pri izboru polinoma. Bolj realen primer tega bomo videli kasneje, ko si bomo ogledali polinome stopnje več kot 2.

Za testiranje družin polinomov sem uporabil preprost program v Matlabu, ki za podana naravna števila n_f, n_p, n_x in s izbere n_f naključnih celoštevilskih polinomov stopnje s, n_p naključnih praštevil v podanem razponu in n_x naključnih začetnih vrednosti. Nato za vsako trojko (polinom p, praštevilo q, začetna vrednost x_z) konstruira zaporedje $(x_n)_n$ kot:

$$x_0 = x_z, \qquad x_n = p(x_{n-1}) \ (mod\ q)$$

Program se ustavi, ko se prvič zgodi, da za neki števili t in s velja: $x_{s+t} = x_s$. Tedaj je zaporedje postalo periodično, s je dolžina predperiode, t pa dolžina periode. Program za vsak naključno izbrani polinom vrne povprečno velikost $s + t$.

Linearni polinomi

Izkaže se, da so polinomi oblike $p(x) = ax + b$ ponavadi slaba izbira. Dolžina cikla, ki ga generira linearni polinom pri praštevilskem modulu q, je enaka redu elementa a v končni grupi $(\mathbb{Z}_q, \cdot)$. Red a deli moč grupe, zato je red a (običajno velik) delitelj števila $q - 1$. Prav lahko se celo zgodi, da je dolžina cikla linearnega polinoma kar q (denimo pri $a = 1$ in $b \neq 0$), kar pa je zelo nezaželjeno, saj je moč pokazati, da je za psevdonaključno zaporedje pričakovana dolžina cikla $O(\sqrt{q})$, kar je dosti boljše.

Eksperimentalni rezultati potrjujejo slabost linearnih polinomov. Tabela 1 prikazuje povprečno vrednost $s + t$, ki jo vrne zgoraj opisano testiranje za praštevila do 10^3, 10^4, 10^5 in 10^6. Linearni polinomi so se v povprečju obnašali zelo podobno, zato sem v tabelo vključil kar povprečno vrednost $s + t$ za vse testirane linearne polinome in jo primerjal z vrednostjo polinoma $x^2 + 1$. Zaradi kompaktnosti bom polinome v tabele vpisoval kot zaporedja koeficientov.

Polinom	p < 100	p < 1000	p < 10000	p < 100000
1, 0, 1	9,89	36,35	121,7	370,4
naključni linearni	61,9	732,5	8257	77338

Tabela 1

Vidno je zelo slabo obnašanje linearnih polinomov.

Kvadratni polinomi

Polinomi oblike $p(x) = ax^2 + bx + c$ se obnašajo mnogo bolje od linearnih. Standardna izbira za Pollardov algoritem sta polinoma $x^2 + 1$ in $x^2 - 1$. Ta dva polinoma namreč v splošnem pričakovano generirata cikle in predperiode dolžine $O(\sqrt{q})$.

Najprej sem poskusil kar naključno z velikim številom kvadratnih polinomov s celimi koeficienti (v resnici je dovolj gledati celo koeficiente $1, 2, \ldots q - 1$ zaradi računanja po modulu q).

Testiral sem veliko število naključnih kvadratnih polinomov in ugotovil, da se v glavnem v povprečju obnašajo podobno. Povprečno ni bilo velikih odstopanj, torej močno boljših ali slabših polinomov, res pa je, da sta bila polinoma $x^2 + 1$ in $x^2 - 1$ vselej med boljšimi, kar pomeni, da je njuna izbira najbrž smiselna.

Tabela 2 prikazuje rezultate testiranj nekaj polinomov za praštevila velikosti do 10^5 (pri večjih praštevilih bi bilo tako obsežno testiranje že preveč zamudno). To je seveda samo vzorec, saj je bilo testiranih polinomov veliko več, vendar so bili rezultati vselej podobni. Pri večjih praštevilih bi bilo tako obsežno testiranje že preveč zamudno.

Polinom	p < 100000
1, 0, 1	370,2
48649, 73008, 46328	429,6
30770, 64159, 52970	366,1
19089, 20645, 15392	398,2
82434, 63613, 25391	393,3
22748, 85548, 34066	406,9
45838, 44390, 66905	403,4
13883, 16422, 86911	379,1
30194, 75576, 64723	393,2
65621, 98485, 73979	374,7

Tabela 2

Odtod sledi, da je brez znanja o deliteljih števila N povsem smotrno vzeti kar standardni polinom x^2+1, saj je praviloma v povprečju med boljšimi polinomi. Vseeno pa so razlike med naključno izbranimi polinomi tako majhne, da se ob neuspehu gotovo zdi smiselno poskusiti še s kakim drugim polinomom.

Zelo prav bi prišlo, če bi imeli kak par polinomov, za katerega bi vedeli, da če so cikli pri danem praštevilu za enega dolgi, so za drugega kratki, vendar žal taki pari niso znani. Motivacija za tako razmišljanje pride iz standarnega para x^2+1 oz. x^2-1 , saj je znano, da so pri nekaterih praštevilih cikli za oba polinoma dolgi.

Polinomi višjih stopenj

Motivacija za uporabo polinomov nizkih stopenj je v tem, da moramo njihovo vrednost izračunati zelo velikokrat in zato želimo za to porabiti karseda majhno število operacij. Da bi torej polinomi višje stopnje upravičili svojo uporabo, bi morali tvoriti ustrezno krajše cikle.
Izkaže se, da vsaj v povprečju ni motiva, da bi uporabljali polinome višjih stopenj. Tabela 3 prikazuje vzorec naključnih polinomov tretje stopnje.

Polinom	p < 100000
2732, 19192, 23211, 7710	424,2
42081, 43730, 67031, 44094	438,5
29917, 19783, 34852,	430,4

88684	
75271, 88895, 90046, 59913	413,8
19025, 51427, 64696, 97614	421,9
83415, 15798, 71934, 27751	422,9
30547, 9437, 18192, 73335	403,1
65639, 38289, 7877, 63926	441,6
2624, 79925, 38433, 24377	358,7
61317, 39621, 67207, 55100	361,4

Tabela 3

Vidno je, da naključni polinomi tretje stopnje ne doprinesejo krajših ciklov od kvadratnih. Enako se izkaže za polinome še višje stopnje, zato je z ozirom na kompleksnejše računanje višanje stopnje nesmiselno, če imamo v mislih naključne polinome.

Ni pa nesmiselno vzeti višje stopnje polinoma, če si s tem lahko pomagamo pri izkoriščanju informacij o deliteljih števila N. Vzemimo za primer K-to Fermatovo število F_K. Vsi njegovi faktorji so kongruentni 1 po modulu 2^{K+2} za $K > 2$, kar lahko uporabimo tako, da za naš polinom izberemo $p(x) = x^{2^{K+2}} + 1$. To pomeni, da bo polinom p generiral psevdonaključen sprehod po vseh za 1 povečanih 2^{K+2}-tih potencah v množici velikosti $q - 1$, teh pa je potemtakem največ $\frac{q-1}{2^{K+2}}$. Zaradi pričakovane velikosti cikla $O(\sqrt{q})$ bo v primerjavi s standardnim polinomom $x^2 + 1$ ta polinom $\sqrt{2^{K+1}}$ –krat hitrejši. Za dovolj velik K se bo torej izplačalo vzeti polinom višje stopnje.

V zvezi s polinomi višjih stopenj omenimo še eno idejo. Če uporabimo interpolacijski polinom (recimo Newtonovo interpolacijo), lahko skonsktruiramo poljubno dolge cikle tako, da enostavno predpišemo ustrezne vrednosti.
Na primer:

$p(x_1) = x_2$
$p(x_2) = x_3$
$p(x_3) = x_4$
$p(x_4) = x_1$

pa smo dobili cikel dolžine 4. Toda v resnici že za interpolacijo teh štirih točk v splošnem potrebujemo polinom stopnje 3. Zdi se, da bo stopnja polinoma naraščala prehitro. V resnici pa je v zvezi s tako konstruiranimi polinomi za Pollardovo ρ metodo malo znanega.

Zaključek

V splošnem, če ne poznamo nobenih lastnosti deliteljev, je torej kvadratni polinom dobra izbira, saj imajo linearni polinomi občutno daljše cikle, polinomi višjih stopenj pa pri naključni izbiri ne prinesejo izboljšanja. Polinoma $x^2 + 1$ in $x^2 - 1$ se ponavadi obneseta dobro, dostikrat pa se izkaže, da je $GCD(x_k - x_u, N) = N$ in tedaj je potrebno poskusiti tudi s kakim drugim.

Polinomi višje stopnje od 2 pa lahko pridejo v poštev, če poznamo kake dodatne lastnosti deliteljev izbranega števila N.

Literatura

[1] M. Petrovič, Metode za razcep števil: diplomsko delo, FMF, 1999

[2] D. M. Bressoud, Factorization and Primality Testing, Springer-Verlag, 1989

[3] N. Koblitz, A Course in Number Theory and Cryptography 2nd edition, Springer-Verlag, 1994

[4] J. Menezes, P. C. Oorschot, S. A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996

[5] R. Crandall, C. Pomerance, Prime Numbers: a Computational Perspective 2nd edition, Springer-Verlag, 2005

[6] D. R. Stinson, Cryptography: theory and practice, 3rd edition, Chapman & Hall/CRC, 2006