Diskrétní matematika B – 5. týden
Aplikace teorie čísel – Počítání s velkými čísly, kryptografie

Michal Bulant

Masarykova univerzita
Fakulta informatiky

jaro 2014

Obsah přednášky

Doporučené zdroje

- Jan Slovák, Martin Panák, Michal Bulant, *Matematika drsně a svižně*, MU Brno, 2013, 774 s. (též jako e-text).
- *Předmětové záložky v IS MU*
- Michal Bulant, výukový text k přednášce **Elementární teorie čísel**, `http://is.muni.cz/el/1431/podzim2012/M6520/um/main-print.pdf`
- William Stein, **Elementary Number Theory: Primes, Congruences, and Secrets**, Springer, 2008. Dostupné na `http://wstein.org/ent/ent.pdf` a `http://wstein.org/edu/2007/spring/ent/`

Doporučené zdroje – algoritmy

- Radan Kučera, výukový text k přednášce **Algoritmy teorie čísel**, `http://www.math.muni.cz/~kucera/texty/ATC10.pdf`
- Donald E. Knuth, **The Art Of Computer Programming**.
- Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. **Introduction to Algorithms**, MIT Press, 2009.
- Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, **Handbook of Applied Cryptography**, CRC Press, 2001. (též jako *e*-text).

Základní úlohy výpočetní teorie čísel

V mnoha praktických úlohách využívajících výsledky teorie čísel je zapotřebí umět rychle provést jeden či více z následujících výpočtů:

1. běžné aritmetické operace (součet, součin, dělení se zbytkem) na celých číslech,
2. zbytek mocniny celého čísla a na přirozené číslo n po dělení daným m.
3. inverzi celého čísla a modulo $m \in \mathbb{N}$,
4. největší společný dělitel dvou celých čísel (a případně koeficienty do Bezoutovy rovnosti),
5. rozhodnout o daném čísle, je-li prvočíslo nebo složené,
6. v případě složenosti rozložit dané číslo na součin prvočísel.

Základní aritmetické operace

Základní aritmetické operace se i na velkých číslech obvykle provádějí obdobně jako jsme se to učili na základní a střední škole, kdy umíme sčítat v *lineárním*, násobit a dělit se zbytkem v *kvadratickém* čase. Pro **násobení**, které je základem mnoha dalších operací, existují asymptoticky rychlejší algoritmy (typu *rozděl a panuj*) - např. první takový Karatsubův (1960) časové náročnosti $\Theta(n^{\log_2 3})$ nebo algoritmus Schönhage-Strassenův (1971) časové náročnosti $\Theta(n \log n \log\log n)$, který využívá tzv. Fast Fourier Transform. Ten je ale přes svou asymptotickou převahu výhodný až pro násobení čísel majících alespoň desítky tisíc cifer (a používá se tak např. v GIMPS).
Pěkný přehled je např. na `http://en.wikipedia.org/wiki/Computational_complexity_of_mathematical_operations`

GCD a modulární inverze

Jak už jsme ukazovali dříve, výpočet řešení kongruence $a \cdot x \equiv 1 \pmod{m}$ s neznámou x lze snadno (díky Bezoutově větě) převést na výpočet největšího společného dělitele čísel a a m a na hledání koeficientů k, l do Bezoutovy rovnosti $k \cdot a + l \cdot m = 1$ (nalezené k je pak onou hledanou inverzí a modulo m).

```
function extended_gcd(a, m)
    if m == 0
        return (1, 0)
    else
        (q, r) := divide (a, m)
        (k, l) := extended_gcd(m, r)
        return (l, k - q * l)
```

Podrobná analýza (viz např. [Knuth] nebo [Wiki]) ukazuje, že tento algoritmus je **kvadratické** časové složitosti.

Modulární umocňování

Modulární umocňování je, jak jsme již viděli dříve, velmi využívaná operace mj. při ověřování, zda je dané číslo prvočíslo nebo číslo složené. Jedním z efektivních algoritmů je tzv. **modulární umocňování zprava doleva**:

```
function modular_pow(base, exponent, modulus)
    result := 1
    while exponent > 0
      if (exponent mod 2 == 1):
        result := (result * base) mod modulus
      exponent := exponent >> 1
      base = (base * base) mod modulus
    return result
```

Algoritmus modulárního umocňování je založen na myšlence, že např. při počítání $2^{64} \pmod{1000}$

- není třeba nejprve počítat 2^{64} a poté jej vydělit se zbytkem číslem 1000, ale lépe je postupně násobit „dvojky“ a kdykoliv je výsledek větší než 1000, provést redukci modulo 1000,
- ale zejména, že není třeba provádět takové množství násobení (v tomto případě 63 naivních násobení je možné nahradit pouze šesti umocněními na druhou, neboť

$$2^{64} = (((((2^2)^2)^2)^2)^2)^2.$$

Příklad (Ukázka průběhu algoritmu)

Vypočtěme $2^{560} \pmod{561}$. Protože $560 = (1000110000)_2$, dostaneme uvedeným algoritmem

exponent	base	result	exp's last digit
560	2	1	0
280	4	1	0
140	16	1	0
70	256	1	0
35	460	1	1
17	103	460	1
8	511	256	0
4	256	256	0
2	460	256	0
1	103	256	1
0	511	1	0

A tedy $2^{560} \equiv 1 \pmod{561}$.

Efektivita modulárního umocňování

V průběhu algoritmu se pro každou binární číslici exponentu provede umocnění základu na druhou modulo n (což je operace proveditelná v nejhůře kvadratickém čase), a pro každou „jedničku“ v binárním zápisu navíc provede jedno násobení. Celkově jsme tedy schopni provést modulární umocňování nejhůře v **kubickém** čase.

Přestože platí základní věta aritmetiky, která nám garantuje, že každé přirozené číslo se dá jednoznačným způsobem rozložit na součin prvočísel, praktické nalezení tohoto rozkladu je obvykle velmi výpočetně náročná operace, obvykle prováděná v několika krocích:

1. nalezení všech dělitelů nepřevyšujících určitou hranici (metodou pokusného dělení všemi prvočísly až do této hranice, typicky je touto hranicí cca 10^6)
2. otestování zbylého faktoru na složenost (tzv. test na složenost, testující některou nutnou podmínku prvočíselnosti)
   a) pokud test složenosti prohlásil, že zkoumané číslo je asi prvočíslo, pak testem na prvočíselnost ověřit, že je to opravdu prvočíslo.
   b) pokud test složenosti prohlásil, že zkoumané číslo je složené, pak nalézt netriviálního dělitele.

Takto je posloupnost kroků prováděna z toho důvodu, že jednotlivé algoritmy mají postupně (výrazně) rostoucí časovou složitost.
V roce 2002 Agrawal, Kayal a Saxena publikovali algoritmus, který testuje prvočíselnost v polynomiálním čase, prakticky je ale zatím stále efektivnější používat výše uvedený postup.

Jak s jistotou poznat složená čísla

Takzvané testy na složenost testují některou nutnou podmínku prvočíselnosti. Nejjednodušší takovou podmínkou je Malá Fermatova věta.

Fermatův test

Existuje-li pro dané N nějaké $a \not\equiv 0 \pmod{N}$ takové, že $a^{N-1} \not\equiv 1 \pmod{N}$, pak N není prvočíslo.

Bohužel nemusí být pro dané složené N snadné najít takové a, že Fermatův test odhalí složenost N; pro některá výjimečná N dokonce jediná taková a jsou ta soudělná s N; jejich nalezení je tedy ekvivalentní s nalezením dělitele, a tedy i s rozkladem N na prvočísla.

Carmichaelova čísla

Skutečně existují taková nehezká (nebo extrémně hezká?) složená čísla N, která splňují, že pro libovolné a nesoudělné s N platí $a^{N-1} \equiv 1 \pmod{N}$. Taková čísla se nazývají Carmichaelova, nejmenší z nich je $561 = 3 \cdot 11 \cdot 17$ a teprve v roce 1992 se podařilo dokázat, že jich je dokonce nekonečně mnoho (v OEIS jde o posloupnost A002997: $561, 1105, 1729, 2465, 2821, \ldots$).

Příklad

Dokážeme, že 561 je Carmichaelovo, tj. že pro každé $a \in \mathbb{N}$, které je nesoudělné s $3 \cdot 11 \cdot 17$, platí $a^{560} \equiv 1 \pmod{561}$.
Z vlastností kongruencí víme, že stačí dokázat tuto kongruenci modulo $3, 11$ i 17. To ale dostaneme přímo z Malé Fermatovy věty, protože takové a splňuje $a^2 \equiv 1 \pmod{3}$, $a^{10} \equiv 1 \pmod{11}$, $a^{16} \equiv 1 \pmod{17}$, přičemž $2, 10$ i 16 dělí 560 (viz též Korseltovo kritérium).

Věta (Korseltovo kritérium)

Složené číslo n je Carmichaelovým číslem, právě když je nedělitelné čtvercem (square-free) a pro všechna prvočísla p dělící n platí $p-1 \mid n-1$.

Příklad

Dokažte, že čísla 2465 a 2821 jsou Carmichaelova.

Eulerův test (též Euler-Jacobi, Solovay-Strassen)

Fermatův test lze zlepšit s využitím kvadratických zbytků na Eulerův test, ale výše zmíněný problém se ani tak zcela neodstraní.

Eulerův test

Je-li N prvočíslo a $a \in \mathbb{Z}, N \nmid a$, pak

$$a^{\frac{N-1}{2}} \equiv (a/N) \pmod{N}.$$

Příklad

Uvažme $a = 5$: [a] Pak $5^{280} \equiv 1 \pmod 3, 5^{280} \equiv 1 \pmod{11}$, přitom $5^{280} \equiv -1 \pmod{17}$, proto určitě $5^{280} \not\equiv \pm 1 \pmod{561}$. Zde došlo k tomu, že neplatilo $a^{\frac{N-1}{2}} \equiv \pm 1 \pmod N$, proto ani nebylo třeba testovat hodnotu Jacobiho symbolu, často ale právě Eulerův test může odhalit složené číslo i v případě, kdy tato mocnina je rovna ± 1.

[a] Testování by selhalo už pro $a = 3$, ale to je dělitel, my chceme ukázat, že test může uspět i bez nalezení dělitele

Příklad

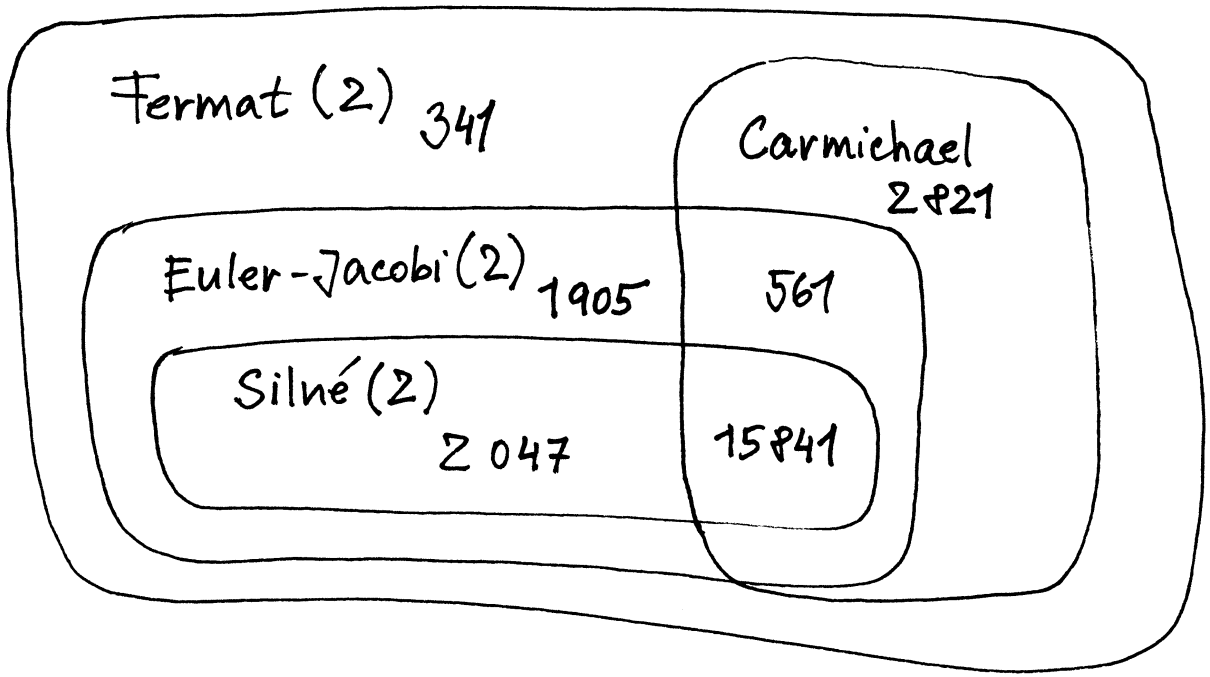
Test, zda $a^{\frac{N-1}{2}} \equiv \pm 1 \pmod{N}$, neodhalí například $N = 1729 = 7 \cdot 13 \cdot 19$, neboť $\frac{N-1}{2} = 864 = 2^5 \cdot 3^3$ je dělitelné 6, 12 i 18 a tedy z Fermatovy věty plyne, že pro všechna celá čísla a nesoudělná s N platí $a^{\frac{N-1}{2}} \equiv 1 \pmod{N}$.
Přitom ale pro $a = 11$ dostaneme $\left(\frac{11}{1729}\right) = -1$ a Eulerův test tedy složenost čísla 1729 odhalí.

Poznamenejme, že hodnotu Legendreova nebo Jacobiho symbolu $\left(\frac{a}{n}\right)$ lze díky zákonu kvadratické reciprocity spočítat v lepším než kubickém čase.

Pseudoprvočísla

Složené číslo n se nazývá **pseudoprvočíslo**, pokud projde testem na složenost a není jím odhaleno jako složené. Máme tak

1. Fermatova pseudoprvočísla o základu a
2. Eulerova pseudoprvočísla
3. silná (strong) pseudoprvočísla o základu a, pokud projdou následujícím testem na složenost.



Test na složenost – zesílení Malé Fermatovy věty

Nechť p je liché prvočíslo. Pišme $p - 1 = 2^t \cdot q$, kde t je přirozené číslo a q je liché. Pak pro každé celé číslo a nedělitelné p buď platí $a^q \equiv 1 \pmod{p}$ nebo existuje $e \in \{0, 1, \ldots, t-1\}$ splňující $a^{2^e q} \equiv -1 \pmod{p}$.

Ukazuje se, že tento snadný test výrazně zesiluje schopnost rozpoznávat složená čísla. Nejmenší silné pseudoprvočíslo o základu 2 je 2047 (přitom nejmenší Fermatovo o základu 2 bylo již 341) a při otestování základů 2,3 a 5 dostaneme nejmenší pseudoprvočíslo 25326001. Jinými slovy, pokud nám stačí testovat pouze čísla do $2 \cdot 10^7$, pak stačí tento test na složenost provést pouze pro základy 2,3 a 5. Pokud číslo není odhaleno jako složené, pak je určitě prvočíslem. Na druhou stranu, bylo dokázáno, že žádná konečná báze není dostatečná.

Test Millera a Rabina

Test Millera a Rabina je praktickou aplikací předchozího testu, kdy jsme navíc schopni omezit pravděpodobnost neúspěchu.

Věta

Nechť $N > 10$ je liché složené číslo. Pišme $N - 1 = 2^t \cdot q$, kde t je přirozené číslo a q je liché. Pak nejvýše čtvrtina z čísel množiny $\{a \in \mathbb{Z};\ 1 \leq a < N,\ (a, N) = 1\}$ splňuje následující podmínku:

$$a^q \equiv 1 \pmod{N}$$

nebo existuje $e \in \{0, 1, \ldots, t-1\}$ splňující

$$a^{2^e q} \equiv -1 \pmod{N}.$$

V praktických implementacích se obvykle testuje cca 20 náhodných základů (příp. nejmenších prvočíselných základů). V takovém případě dostáváme z předchozí věty, že pravděpodobnost neodhalení složeného čísla je menší než 2^{-40}.
Časová náročnost algoritmu je asymptoticky stejná jako složitost modulárního umocňování, tedy nejhůře kubická. Je ale třeba si uvědomit, že test je nedeterministický a spolehlivost jeho deterministické verze závisí na tzv. zobecněné Riemannově hypotéze (GRH).

Testy na prvočíselnost

Testy na prvočíselnost přicházejí na řadu obvykle ve chvíli, kdy testy na složenost prohlásí, že jde *pravděpodobně o prvočíslo*, případně se provádějí rovnou u speciálních typů čísel. Uveďme nejprve přehled nejznámějších testů.

1. AKS (2002) – obecný polynomiální test na prvočísla
2. Pocklington-Lehmerův test – test na prvočíselnost subexponenciální složitosti
3. Lucas-Lehmerův test – test prvočíselnosti pro Mersenneho čísla
4. Pépinův test (1877) – test prvočíselnosti pro Fermatova čísla
5. ECPP - test prvočíselnosti založený na tzv. eliptických křivkách

Speciální testy – Mersenneho čísla

Lucas-Lehmerův test

Definujme posloupnost $(s_n)_{n=0}^{\infty}$ rekurzívně předpisem $s_0 = 4, s_{n+1} = s_n^2 - 2$.
Pak je číslo $M_p = 2^p - 1$ prvočíslo, právě tehdy, když M_p dělí s_{p-2}.

```
// Determine if M_p = 2^p - 1 is prime
Lucas-Lehmer(p)
    var s = 4
    var M = 2^p - 1
    repeat p - 2 times:
        s = s^2 - 2 (mod M)
    if s = 0 return PRIME else return COMPOSITE
```

Časová složitost testu je asymptoticky stejná jako v případě Miller-Rabinova testu, v konkrétních případech je ale efektivnější.

Speciální testy – Fermatova čísla

Fermatova čísla jsou čísla tvaru $F_n = 2^{2^n} + 1$. Pierre de Fermat v 17. století vyslovil hypotézu, že všechna čísla tohoto tvaru jsou prvočísly (zřejmě veden snahou zobecnit pozorování pro $F_0 = 3, F_1 = 5, F_2 = 17, F_3 = 257$ a $F_4 = 65537$. V 18. století ale Leonhard Euler zjistil, že $F_5 = 641 \times 6700417$ a dodnes se nepodařilo nalézt žádné další Fermatovo prvočíslo. Vzhledem k rychle rostoucí velikosti těchto čísel je počítání s nimi velmi časově náročné (a ani následující test tak není příliš používán). V současné době nejmenší netestované Fermatovo číslo je F_{33}, které má 2 585 827 973 číslic a je tak výrazně větší než největší dosud nalezené prvočíslo.

Pépinův test

Označme $F_n = 2^{2^n} + 1$ tzv. n-té Fermatovo číslo. Pak F_n je prvočíslo, právě když

$$3^{\frac{F_n-1}{2}} \equiv -1 \pmod{F_n}.$$

Vidíme, že jde o velmi jednoduchý test, který je vlastně pouze malou částí Eulerova testu na složenost.

Důkaz korektnosti Pépinova testu.

Platí-li $3^{\frac{F_n-1}{2}} \equiv -1 \pmod{F_n}$, je nutně $F_n - 1$ řádem čísla 3 modulo F_n, proto je F_n prvočíslo.
Obráceně, nechť je F_n prvočíslo. Z Eulerova kritéria dostáváme, že $3^{\frac{F_n-1}{2}} \equiv \left(\frac{3}{F_n}\right) \pmod{F_n}$, tj. stačí nám určit hodnotu $\left(\frac{3}{F_n}\right)$. To je ale snadné, protože $F_n \equiv 2 \pmod 3$ a tedy $\left(\frac{F_n}{3}\right) = -1$. Dále $F_n \equiv 1 \pmod 4$, proto díky zákonu kvadratické reciprocity dostáváme $\left(\frac{3}{F_n}\right) = -1$. □

Pocklington-Lehmerův test

Na závěr uveďme i obecný test na prvočíselnost, který použijeme, pokud chceme vysokou pravděpodobnost Miller-Rabinova algoritmu proměnit v jistotu (ta jistota je ale relativní – udává se, že pravděpodobnost selhání Miller-Rabinova algoritmu je nižší než HW chyba během výpočtu).

Věta

Nechť N je přirozené číslo, $N > 1$. Nechť p je prvočíslo dělící $N - 1$. Předpokládejme dále, že existuje $a_p \in \mathbb{Z}$ tak, že

$$a_p^{N-1} \equiv 1 \pmod{N} \quad a \quad \left(a_p^{\frac{N-1}{p}} - 1, N \right) = 1.$$

Nechť p^{α_p} je nejvyšší mocnina p dělící $N - 1$. Pak pro každý kladný dělitel d čísla N platí

$$d \equiv 1 \pmod{p^{\alpha_p}}.$$

Důkaz věty Pocklingtona a Lehmera.

Každý kladný dělitel d čísla N je součinem prvočíselných dělitelů čísla N, větu dokažme pouze pro d prvočíslo. Podle Fermatovy věty platí $a_p^{d-1} \equiv 1 \pmod{d}$, neboť $(a_p, N) = 1$. Protože $\left(a_p^{\frac{N-1}{p}} - 1, N\right) = 1$, platí $a_p^{\frac{N-1}{p}} \not\equiv 1 \pmod{d}$.
Označme e řád a_p modulo d. Pak platí $e \mid d-1$, $e \mid N-1$ a $e \nmid \frac{N-1}{p}$. Kdyby $p^{\alpha_p} \nmid e$, z $e \mid N-1$ by plynulo $e \mid \frac{N-1}{p}$, spor. Je tedy $p^{\alpha_p} \mid e$, a tedy i $p^{\alpha_p} \mid d-1$. □

Užití věty Pocklingtona a Lehmera

Tvrzení

Nechť $N \in \mathbb{N}$, $N > 1$. Předpokládejme, že můžeme psát $N - 1 = F \cdot U$, kde $(F, U) = 1$ a $F > \sqrt{N}$, přičemž známe rozklad čísla F na prvočinitele. Pak platí:

- *jestliže pro každé prvočíslo $p \mid F$ můžeme najít $a_p \in \mathbb{Z}$ z předchozí věty, pak je N prvočíslo;*
- *je-li N prvočíslo, pak pro libovolné prvočíslo $p \mid N - 1$ existuje $a_p \in \mathbb{Z}$ s požadovanými vlastnostmi.*

Důkaz.

ad 1. Podle Věty je $d \equiv 1 \pmod{p^{\alpha_p}}$ pro všechny prvočíselné faktory F, proto je $d \equiv 1 \pmod{F}$, a tedy $d > \sqrt{N}$.
ad 2. Stačí za a_p zvolit primitivní kořen modulo prvočíslo N (nezávisle na p). □

Příklad

Předchozí test v sobě zahrnuje Pépinův test (totiž pro $N = F_n$ máme $p = 2$, kterému vyhovuje svědek prvočíselnosti $a_p = 3$).

Hledání dělitele

Máme-li testem na složenost potvrzeno, že jde o číslo složené, obvykle chceme najít netriviálního dělitele. Jde ale o výrazně obtížnější úkol (což je na druhé straně výhodné pro RSA a podobné protokoly), proto si k tématu uvedeme jen stručný přehled používaných metod.

1. Pokusné dělení
2. Pollardova ρ-metoda
3. Pollardova $p-1$ metoda
4. faktorizace pomocí eliptických křivek
5. Metoda kvadratického síta (QS)
6. Metoda síta v číselném tělesa (NFS)

Podrobnosti viz `M8190` *Algoritmy teorie čísel*.

Pollardova rho-metoda – ukázka

Pollardova ρ-metoda je algoritmus speciálně vhodný pro hledání *relativně* malých dělitelů, založená na myšlence, že pro náhodnou funkci $f : S \to S$, kde S je konečná n-prvková množina, se musí posloupnost $(x_n)_{n=0}^{\infty}$, kde $x_{n+1} = f(x_n)$ zacyklit. Přitom předperioda i perioda má očekávanou délku $\sqrt{\pi \cdot n/8}$.

Inputs: n, the integer to be factored;
and $f(x)$, a pseudo−random function
Output: a non−trivial factor of n, or failure.
$x := 2, y := 2; d := 1$
While $d = 1$ do
$x := f(x)$
$y := f(f(y))$
$d := GCD(|x - y|, n)$
If $d = n$, return failure.
Else, return d.

Příklad

Nalezněte dělitele čísla 455459.

Uvažujme funkci $f(x) = x^2 + 1$ (mlčky předpokládáme, že se tato funkce modulo neznámý prvočíselný dělitel p čísla n chová náhodně a má tak požadované vlastnosti) a v jednotlivých iteracích počítáme $a \leftarrow f(a) \pmod{n}$, $b \leftarrow f(f(b)) \pmod{n}$.

a	b	d
5	26	1
26	2871	1
677	179685	1
2871	155260	1
44380	416250	1
179685	43670	1
121634	164403	1
155260	247944	1
44567	68343	743

Kryptografie s veřejným klíčem (PKC)

Dva hlavní úkoly pro PKC jsou zajistit

- šifrování, kdy zprávu **zašifrovanou** veřejným klíčem příjemce není schopen rozšifrovat nikdo kromě něj (resp. držitele jeho soukromého klíče)
- podepisování, kdy integrita zprávy **podepsané** soukromým klíčem odesílatele může být ověřena kýmkoliv s přístupem k veřejnému klíči odesílatele

Nejčastěji používané systémy PKC:

- RSA (šifrování) a odvozený systém pro podepisování zpráv
- Digital signature algorithm (DSA) a varianta založená na eliptických křivkách (ECDSA)
- Rabinův kryptosystém (a podepisování)
- ElGamal kryptosystém (a podepisování)
- Kryptografie eliptických křivek (ECC)
- Diffie-Hellmanův protokol na výměnu klíčů (DH)

Princip digitálního podpisu

Podepisování

1. Vygeneruje se otisk (hash) H_M zprávy pevně stanovené délky (např. 160 nebo 256 bitů).
2. Podpis zprávy $S_A(H_M)$ je vytvořen (pomocí dešifrování) z tohoto hashe s nutností znalosti soukromého klíče podepisujícího.
3. Zpráva M (případně zašifrovaná veřejným klíčem příjemce) je spolu s podpisem odeslána.

Ověření podpisu

1. K přijaté zprávě M se (po jejím případném dešifrování) vygeneruje otisk H'_M
2. S pomocí veřejného klíče (deklarovaného) odesílatele zprávy se rekonstruuje původní otisk zprávy $V_A(S_A(H_M)) = H_M$.
3. Oba otisky se porovnají $H_M = H'_M$?.

(Kryptografická) hashovací funkce

(Kryptografická) hashovací funkce má mít následující vlastnosti:

- Pro libovolnou zprávu je snadné nalézt její hash.
- Nelze (v reálném čase) zjistit zprávu s požadovaným hashem.
- Nelze (v reálném čase) nalézt dvě zprávy se stejným hashem.
- Každá změna zprávy se projeví změnou hashe.

Příklady:

- MD5 (128 bit, Rivest 1992) – není collision-resistant
- SHA-1 (160 bit, NSA 1995) – od roku 2005 považována za nedostatečně odolnou proti kolizím
- RIPEMD-320
- SHA-3

RSA

Ron Rivest, Adi Shamir, Leonard Adleman (1977; C. Cocks,GCHQ – 1973)

- každý účastník A potřebuje dvojici klíčů – veřejný V_A a soukromý S_A
- generování klíčů: zvolí dvě velká prvočísla p, q, vypočte $n = pq$, $\varphi(n) = (p-1)(q-1)$ [n je veřejné, ale $\varphi(n)$ nelze snadno spočítat]
- zvolí **veřejný klíč** e a ověří, že $(e, \varphi(n)) = 1$
- např. pomocí Euklidova algoritmu spočítá **tajný klíč** d tak, aby $e \cdot d \equiv 1 \pmod{\varphi(n)}$
- zašifrování numerického *kódu* zprávy M: $C = C_e(M) \equiv M^e \pmod{n}$
- dešifrování šifry C: $OT = D_d(C) \equiv C^d \pmod{n}$

Poznámka

- Korektní naprogramování bez postranních kanálů není triviální (viz např. PKCS#1, RFC 3447).
- Analogicky podepisování (hashů) zpráv (viz např. DSA)
- Viz RSA factoring challenge (např. rozklad 212 ciferného čísla RSA-704 by býval vynesl 30 000 USD).

Bezpečnost RSA

Bezpečnost RSA je testována od vzniku šifry v roce 1977 a dosud se (s výjimkou postranních kanálů či některých singulárních klíčů) nepodařilo objevit výraznou slabinu (při použití dostatečně velkého klíče, nyní se doporučuje 2048 bitů). Přesto se dodnes neumí dokázat, že problém RSA **skutečně** závisí na nesnadnosti faktorizace.

Požadavky na bezpečnou volbu klíče:

- d je dostatečně velké (viz Wienerův útok)
- p a q nejsou příliš blízká (viz následující příklad)
- není znám útok proti malému veřejnému klíči e, doporučuje se alespoň $e = 65537$.

Útok na blízká p a q

Pomocí tzv. Fermatovy faktorizace se můžeme pokusit rozložit $n = p \cdot q$, pokud máme důvod se domnívat, že rozdíl p a q je malý. Pak totiž

$$n = \left(\frac{p+q}{2}\right)^2 - \left(\frac{p-q}{2}\right)^2,$$

kde $s = (p-q)/2$ je malé a $t = (p+q)/2$ je pouze o málo větší než $\sqrt{n}$. Stačí tedy testovat čísla $t = \lceil\sqrt{n}\rceil, t = \lceil\sqrt{n}\rceil + 1, t = \lceil\sqrt{n}\rceil + 2$, dokud nebude $t^2 - n$ druhou mocninou (což lze efektivně testovat).

Rabinův kryptosystém

Prvním veřejným kryptosystémem, k jehož prolomení je prokazatelně potřeba faktorizovat modul, je **Rabinův kryptosystém**, který si uvedeme ve zjednodušené verzi:

- každý účastník A potřebuje dvojici klíčů – veřejný V_A a soukromý S_A
- generování klíčů: A zvolí dvě podobně velká prvočísla $p, q \equiv 3 \pmod 4$, vypočte $n = pq$.
- $V_A = n$, $S_A = (p, q)$
- zašifrování numerického *kódu* zprávy M: $C = C_e(M) \equiv M^2 \pmod n$
- dešifrování šifry C: vypočtou se (čtyři) odmocniny z C modulo n a snadno se otestuje, která z nich byla původní zprávou.

Výpočet druhé odmocniny z C modulo $n = pq$, kde $p \equiv q \equiv 3 \pmod 4$

- vypočti $r = C^{(p+1)/4} \pmod p$ a $s = C^{(q+1)/4} \pmod q$
- vypočti a, b tak, že $ap + bq = 1$
- polož[a] $x = (aps + bqr) \pmod n$, $y = (aps - bqr) \pmod n$
- druhými odmocninami z C modulo n jsou $\pm x$, $\pm y$.

[a] Uvědomte si, že jde vlastně o aplikaci Čínské zbytkové věty!

Příklad

V Rabinově kryptosystému Alice zvolila za svůj soukromý klíč $p = 23, q = 31$, veřejným klíčem je pak $n = pq = 713$. Zašifrujte zprávu $m = 327$ pro Alici a ukažte, jak bude Alice tuto zprávu dešifrovat.

Řešení

$c = 692$, kandidáti původní zprávy jsou $\pm 4 \cdot 23 \cdot 14 \pm 3 \cdot 31 \cdot 18 \pmod{713}$.

Diffie-Hellman key exchange

Whitfield Diffie, Martin Hellman (1976; M. Williamson, GCHQ - 1974)
Výměna klíčů pro symetrickou kryptografii bez předchozího kontaktu (tj. náhrada jednorázových klíčů, kurýrů s kufříky, . . .).

- Dohoda stran na **prvočísle** p a primitivním kořenu g modulo p (veřejné)
- Alice vybere náhodné a a pošle $g^a \pmod{p}$
- Bob vybere náhodné b a pošle $g^b \pmod{p}$
- Společným klíčem pro komunikaci je $g^{ab} \pmod{p}$.

Poznámka

- Problém diskrétního logaritmu (DLP)
- Nezbytná autentizace (*man in the middle attack*)

Kryptosystém ElGamal

Z protokolu DH na výměnu klíčů odvozen šifrovací algoritmus ElGamal:

- Alice zvolí prvočíslo p spolu s primitivním kořenem g
- Alice zvolí **tajný klíč** x, spočítá $h = g^x \pmod{p}$ a zveřejní **veřejný klíč** (p, g, h)
- šifrování zprávy M: Bob zvolí náhodné y a vypočte $C_1 = g^y \pmod{p}$ a $C_2 = M \cdot h^y \pmod{p}$ a pošle (C_1, C_2)
- dešifrování zprávy: $OT = C_2 / C_1^x$

Poznámka

Analogicky jako v případě RSA lze odvodit podepisování.

Eliptické křivky

Eliptické křivky jsou rovinné křivky o rovnici tvaru $y^2 = x^3 + ax + b$ a zajímavé jsou tím, že na jejich bodech lze definovat operace tak, že výslednou strukturou bude komutativní grupa.
Přitom uvedené operace lze efektivně provádět a navíc se ukazuje, že mají (nejen) pro kryprografii zajímavé vlastnosti – srovnatelné bezpečnosti jako RSA lze dosáhnout již s podstatně kratšími klíči.
Výhodou je rovněž velké množství použitelných eliptických křivek (a tedy grup různé struktury) podle volby parametru a, b .
Protokoly:

- ECDH - přímá varianta DH na eliptické křívce (jen místo generátoru se vybere *vhodný* bod na křivce)
- ECDSA - digitální podpis pomocí eliptických křivek.

Poznámka

Problém diskrétního logaritmu (ECDLP).
Navíc se ukazuje, že eliptické křivky jsou velmi dobře použitelné při faktorizaci prvočísel.

Kódování

Při přenosu informace zpravidla dochází k její deformaci. Budeme pro jednoduchost pracovat s modelem, kdy jednotlivé částečky informace jsou buď nuly nebo jedničky (tj. prvky v $\mathbb{Z}_2$) a přenášíme slova o k bitech. Obdobné postupy jsou možné i nad ostatními konečnými tělesy.
Přenosové chyby chceme

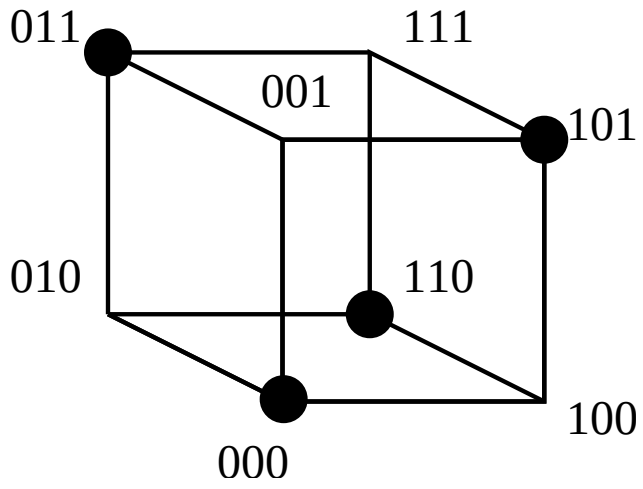
1. rozpoznávat
2. opravovat

a za tím účelem přidáváme dodatečných $n-k$ bitů informace pro pevně zvolené $n > k$.
Všech slov o k bitech je 2^k a každé z nich má jednoznačně určovat jedno **kódové slovo** z 2^n možných. Máme tedy ještě

$$2^n - 2^k = 2^k(2^{n-k} - 1)$$

slov, která jsou chybová. Lze tedy tušit, že pro veliké k nám i malý počet přidaných bitů (tj. $n-k$) dává hodně redundantní informace.

Úplně jednoduchým příkladem je **kód kontrolující paritu**. Kódové slovo o $k+1$ bitech je určené tak, aby přidáním prvního bitu byl zaručen sudý počet jedniček ve slově.
Pokud při přenosu dojde k lichému počtu chyb, přijdeme na to.
Dvě různá kódová slova se při tomto kódu vždy liší alespoň ve dvou pozicích, chybové slovo se ale od dvou různých (vhodných) kódových slov liší pouze v pozici jedné. Nemůžeme proto umět chyby opravovat, ani kdybychom věděli, že došlo k právě jedné chybě.
Navíc neumíme detekovat tak obvyklé chyby, jako je záměna dvou sousedních bitů ve slově.



Definice

Hammingova vzdálenost dvou slov je rovna počtu bitů, ve kterých se liší.

Věta

1. *Kód odhaluje r a méně chyb právě, když je minimální Hammingova vzdálenost kódových slov právě $r + 1$.*
2. *Kód opravuje r a méně chyb právě, když je minimální Hammingova vzdálenost kódových slov právě $2r + 1$.*

Jak konstruovat kódová slova, abychom je snadno rozpoznali?
Kontrolu parity jsme už viděli, další triviální možnost je prosté opakování bitů – např. $(3,1)$–kód bere jednotlivé bity a posílá je třikrát po sobě.
Systematickou cestou je pak využití dělitelnosti polynomů. Zpráva $b_0 b_1 \dots b_{k-1}$ je reprezentována jako polynom $m(x) = b_0 + b_1 x + \cdots + b_{k-1} x^{k-1}$.

Definice

Nechť $p(x) = a_0 + a_1 x + \cdots + a_{n-k} x^{n-k} \in \mathbb{Z}_2[x]$ je polynom s $a_0 = 1$, $a_{n-k} = 1$. **Polynomiální kód generovaný polynomem** $p(x)$ je (n,k)–kód jehož slova jsou polynomy stupně menšího než n dělitelné $p(x)$.
Zpráva $m(x)$ je zakódována jako $v(x) = r(x) + x^{n-k} m(x)$, kde $r(x)$ je zbytek po dělení polynomu $x^{n-k} m(x)$ polynomem $p(x)$.

Z definice víme, že

$$v(x) = x^{n-k}m(x) + r(x) = q(x)p(x) + r(x) + r(x) = q(x)p(x).$$

Budou tedy všechna kódová slova dělitelná $p(x)$.
Původní zpráva je obsažena přímo v polynomu $v(x)$, takže dekódování správného slova je snadné.

Příklad

1. Polynom $p(x) = 1 + x$ generuje $(n, n-1)$–kód kontroly parity pro všechna $n \geq 3$.
2. Polynom $p(x) = 1 + x + x^2$ generuje $(3, 1)$–kód opakování bitů.

První tvrzení plyne z toho, že $1 + x$ dělí polynom $v(x)$ tehdy a jen tehdy, když $v(1) = 0$ a to nastane tehdy, když je ve $v(x)$ sudý počet nenulových koeficientů. Druhé je zřejmé.

Přenos slova $v \in \mathbb{Z}_2^n$ dopadne příjmem polynomu

$$u(x) = v(x) + e(x)$$

kde $e(x)$ je tzv. **chybový polynom** reprezentující vektor chyby přenosu.
Chyba je rozpoznatelná pouze, když generátor kódu $p(x)$ nedělí $e(x)$. Máme proto zájem o polynomy, které nevystupují jako dělitelé zbytečně často.

Definice

Ireducibilní polynom $p(x) \in \mathbb{Z}_2[x]$ stupně m se nazývá **primitivní**, jestliže $p(x)$ dělí polynom $(x^k - 1)$ pro $k = 2^m - 1$, ale nedělí jej pro žádná menší k.

Věta

Je-li $p(x)$ primitivní polynom stupně m, pak pro všechna $n \leq 2^m - 1$ rozpoznává příslušný $(n, n-m)$–kód všechny jednoduché a dvojité chyby.

Důsledek

Je-li $q(x)$ primitivní polynom stupně m, pak pro všechna $n \leq 2^m - 1$ rozpoznává $(n, n-m-1)$–kód generovaný polynomem $p(x) = q(x)(1+x)$ všechny dvojité chyby a všechna slova s lichým počtem chyb.

Tabulka dává o informace o výsledcích předchozích dvou vět pro několik polynomů:

primitivní polynom	kontrolní bity	délka slova
$1 + x + x^2$	2	3
$1 + x + x^3$	3	7
$1 + x + x^4$	4	15
$1 + x^2 + x^5$	5	31
$1 + x + x^6$	6	63
$1 + x^3 + x^7$	7	127
$1 + x^2 + x^3 + x^4 + x^8$	8	255
$1 + x^4 + x^9$	9	511
$1 + x^3 + x^{10}$	10	1023

Definice

Lineární kód je injektivní lineární zobrazení $g : (\mathbb{Z}_2)^k \to (\mathbb{Z}_2)^n$. Matice G typu k/n reprezentující toto zobrazení v standardních bazích se nazývá generující **matice kódu**.

Pro každé slovo v je

$$u = G \cdot v$$

příslušné kódové slovo.

Věta

Každý polynomiální (n, k)–kód je lineární kód.

Generující matice $(7, 4)$ kódu příslušná k polynomu $p(x) = 1 + x^2 + x^3$ je

$$G = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Věta

Je-li g lineární kódování s maticí

$$G = \begin{pmatrix} P \\ \mathbb{E}_k \end{pmatrix},$$

potom zobrazení $h : (\mathbb{Z}_2)^n \to (\mathbb{Z}_2)^k$ s maticí

$$H = \begin{pmatrix} \mathbb{E}_{n-k} & P \end{pmatrix}$$

má následující vlastnosti

1. $\operatorname{Ker} h = \operatorname{Im} g$, *tj.*
2. *přijaté slovo u je kódové slovo právě, když je $H \cdot u = 0$*

Matici H z věty se říká **matice kontroly parity** příslušného (n, k)–kódu.

Samoopravné kódy

Jak jsme viděli, přenos zprávy u dává výsledek

$$v = u + e.$$

To je ale nad $\mathbb{Z}_2$ ekvivalentní s $e = u + v$.
Pokud tedy známe podprostor $V \subset (\mathbb{Z}_2)^n$ správných kódových slov, víme u každého výsledku, že správné slovo (s opravenými případnými chybami) je ve třídě rozkladu $v + V$ v prostoru $(\mathbb{Z}_2)^n/V$.
Zobrazení (homomorfismus grup) $h : (\mathbb{Z}_2)^n \to (\mathbb{Z}_2)^{n-k}$ má V za jádro, proto indukuje injektivní lineární zobrazení $h : (\mathbb{Z}_2)^n/V \to (\mathbb{Z}_2)^{n-k}$. Jeho hodnoty jsou jednoznačně určeny hodnotami $H \cdot u$.

Definice

Hodnota $H \cdot u$ se nazývá **syndrom** slova u.

Věta

Dvě slova jsou ve stejné třídě rozkladu právě tehdy, když mají týž syndrom.

Samoopravné kódy lze konstruovat tak, že pro každý syndrom určíme prvek v příslušné třídě, který je nejvhodnějším slovem.

Příklad

Buď dán $(6, 3)$ kód nad $\mathbb{Z}_2$ generovaný polynomem $x^3 + x^2 + 1$.

1. Určete jeho generující matici a matici kontroly parity.
2. Dekódujte zprávu 111101 předpokládáte-li, že při přenosu došlo k nejmenšímu možnému počtu chyb.

Řešení

Protože se jedná o lineární kód, stačí určit jak se zakódují bázové vektory 1, x a x^2, tedy určit zbytky polynomů x^3, x^4 a x^5 po dělení polynomem $x^3 + x^2 + 1$. Máme

$$\begin{aligned} x^3 &\equiv x^2 + 1 \pmod{x^3 + x^2 + 1} \\ x^4 &= x(x^3) \equiv x(x^2 + 1) \equiv x^2 + x + 1 \pmod{x^3 + x^2 + 1} \\ x^5 &= x(x^4) \equiv x(x^2 + x + 1) \equiv x + 1 \pmod{x^3 + x^2 + 1} \end{aligned}$$

Řešení (pokr.)

Bázové vektory (zprávy) 100, 010 a 001 se tedy zakódují do vektorů (kódů) 101100, 111010 a 110001, generující matice, resp. matice kontroly parity kódu jsou tedy

$$G = \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{resp.} \quad H = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

Vynásobíme-li přijatou zprávu 111101 maticí kontroly parity, dostáváme syndrom 100 a víme, že při přenosu došlo k chybě. Sestavme tabulku všech syndromů a jim odpovídajících kódových slov.

Syndrom	(Kódová) slova s daným syndromem							
000	**000000**	110001	111010	101100	010110	001011	011101	100111
001	**001000**	111001	110010	100100	011110	000011	010101	101111
010	**010000**	100001	101010	111100	000110	011011	001101	110111
100	**100000**	010001	011010	001100	110110	101011	111101	000111
011	**011000**	101001	100010	110100	001110	010011	**000101**	111111
101	101000	011001	010010	**000100**	111110	100011	110101	001111
110	110000	**000001**	001010	011100	100110	111011	101101	010111
111	111000	001001	**000010**	010100	101110	110011	100101	011111

Řešení (dokončení)

Syndrom 000 mají všechna kódová slova. Počínaje druhým řádkem, je každý řádek tabulky afinním prostorem, jehož zaměřením je vektorový prostor daný prvním řádkem. Zejména je tedy rozdíl každých dvou slov ve stejném řádku nějakým kódovým slovem. Všechna slova s daným syndromem dostaneme přičtením syndromu (doplněného nulami na délku kódového slova) ke všem kódovým slovům. Tzv. vedoucím reprezentantem třídy (řádku, afinního prostoru) odpovídající danému syndromu je slovo s nejmenším počtem jedniček v řádku, a tedy i nejmenším počtem bitových změn, jež je třeba provést, abychom dostali kódové slovo – v našem případě jde o slovo 100000 a jeho odečtením od obdrženého slova dostaneme platné kódové slovo 011101. Je to platné kódové slovo s nejmenší Hammingovou vzdáleností od obdrženého slova. Původní zpráva tedy byla 101.