Čínská věta o zbytcích
RSA

Matematické algoritmy (11MA)

Miroslav Vlček, Jan Přikryl

4. přednáška 11MA
čtvrtek 21. října 2010

verze: 2010-10-21 09:50

1 Čínská věta o zbytcích

Čínská věta o zbytcích *(Chinese Reminder Theorem, CRT)*

Více vzájemně ekvivalentních tvrzení z algebry a teorie čísel. Nejstarší zmínka z Číny ve 3. století našeho letopočtu.

Motivace: Jak najít x takové, že

$$x \equiv 2 \pmod 3,$$
$$x \equiv 3 \pmod 5,$$
$$x \equiv 2 \pmod 7?$$

Čínská věta o zbytcích Postup řešení (1/2)

Zbytkové třídy jsou $[2]_3$, $[3]_5$ a $[2]_7$.

V prvním kroku hledáme nulové a jednotkové zbytkové třídy pro kombinace původních modulů. V našem případě platí

$$\kappa_1 = 70 \equiv 0 \pmod{5 \cdot 7} \quad \wedge \quad \kappa_1 = 70 \equiv 1 \pmod 3,$$
$$\kappa_2 = 21 \equiv 0 \pmod{3 \cdot 7} \quad \wedge \quad \kappa_2 = 21 \equiv 1 \pmod 5,$$
$$\kappa_3 = 15 \equiv 0 \pmod{3 \cdot 5} \quad \wedge \quad \kappa_3 = 15 \equiv 1 \pmod 7.$$

Čínská věta o zbytcích Postup řešení (2/2)

Řešením dané soustavy kongruencí je v takovém případě číslo

$$\hat{x} = 2 \cdot 70 + 3 \cdot 21 + 2 \cdot 15 = 233.$$

Minimální hodnota x je dána třídou kongruence modulo $3 \cdot 5 \cdot 7 = 105$, tedy

$$x = 233 \bmod 105 = 23.$$

1.1 Vlastní tvrzení

Čínská věta o zbytcích Vlastní tvrzení

Nechť $n_1, n_2, \ldots, n_k$ jsou navzájem nesoudělná přirozená čísla, $n_i \geq 2$ pro všechna $i = 1, \ldots, k$. Potom řešení soustavy rovnic

$$\begin{aligned} x &\equiv a_1 \pmod{n_1} \\ x &\equiv a_2 \pmod{n_2} \\ &\vdots \\ x &\equiv a_k \pmod{n_k} \end{aligned}$$

existuje a je určeno jednoznačně v modulo $n = n_1 \cdot n_2 \cdot \ldots \cdot n_k$.

Jak si Sun Tzu ušetří práci Lehký náznak důkazu

Díky nesoudělnosti existuje ve třídě operací modulo n_i ke každému $N_i = n/n_i$ jeho multiplikativní inverze M_i, tedy

$$M_i \cdot N_i \equiv 1 \pmod{n_i}$$

a platí

$$x = \sum_{i=1}^{k} a_i M_i N_i.$$

Ve výše uvedeném případě se zbytkovými třídami $[2]_3$, $[3]_5$ a $[2]_7$ je

$$x = 2 \cdot 2 \cdot 35 + 3 \cdot 1 \cdot 21 + 2 \cdot 1 \cdot 15 = 233.$$

Praktický význam věty

Výpočty modulo velké M lze převést na výpočty modulo menší součinitelé čísla M – zrychlení výpočtu.

Lze generalizovat pro soudělná čísla.

Význam hlavně v šifrovacích systémech.

1.2 Problém nůše s vejci

Problém nůše s vejci Ilustrace použití CRT

V nůši je v vajec. Pokud z ní odebíráme vejce po dvou, třech a pěti najednou, v nůši nakonec zůstane 1, 2, respektive 4 vejce. Pokud odebíráme vejce po sedmi kusech, v nůši nakonec nezůstane vejce žádné.

Jaká je nejmenší hodnota v pro niž může uvedená situace nastat?

Problém nůše s vejci Ilustrace použití CRT (2)

Zbytkové třídy jsou $[1]_2$, $[2]_3$, $[4]_5$ a $[0]_7$.

Hledáme řešení soustavy

$$\begin{aligned} v &\equiv 1 \pmod{2} \\ v &\equiv 2 \pmod{3} \\ v &\equiv 4 \pmod{5} \\ v &\equiv 0 \pmod{7} \end{aligned}$$

Výsledek bude nějaká třída kongruence modulo 210.

Problém nůše s vejci Ilustrace použití CRT (3)

Pro jednotlivé ekvivalence máme

i	n_i	N_i	M_i	a_i
1	2	105	1	1
2	3	70	1	2
3	5	42	3	4
4	7	30	4	0

$$\begin{aligned} v &= (1 \cdot 1 \cdot 105 + 2 \cdot 1 \cdot 70 + 4 \cdot 3 \cdot 42 + 0 \cdot 4 \cdot 30) \bmod 210 \\ &= (105 + 140 + 504 + 0) \bmod 210 = 749 \bmod 210 = 119 \end{aligned}$$

2 Šifrování

2.1 Symetrické a asymetrické šifry

Šifrování Symetrické a asymetrické šifry

Existují dvě základní skupiny šifrovacích algoritmů:

- **Symetrické šifry** u nichž se ten samý klíč používá jak k šifrování, tak i k dešifrování zprávy. Odesílatel i příjemce musí mít k dispozici identické klíče. Příkladem je DES, 3DES, AES.
- **Asymetrické šifry** u nichž se šifruje jiným klíčem, než je klíč určený k dešifrování. Odesílatel po zašifrování již nemá možnost zprávu dešifrovat. Příkladem je RSA (PGP), GnuPG, ElGamal.

Symetrické šifry jsou při stejné délce šifrovacího klíče výrazně bezpečnější, než šifry asymetrické ...

2.2 Výměna klíčů

Diffieho-Hellmanova výměna klíčů Jak se dohodnout na klíči přes nezabezpečený kanál

... ale symetrické šifrování má *základní problém*: distribuci klíčů.

Diffie a Hellman, 1976

Alice a Bob se na klíči mohou dohodnout přes nezabezpečený komunikační kanál. Je pouze třeba zajistit, aby operace, jež Alice a Bob provádějí, *nebyly výpočetně snadno invertovatelné.*

Diffieho-Hellmanova výměna klíčů

Veřejně známé prvočíslo p a $\alpha \in \{2, \ldots, p-2\}$. Oba jako klíč použijí $\alpha^{xy} \bmod p$ – Alice si vymyslí veliké $x \in \mathbb{N}$ a Bobovi pošle $\alpha^x \bmod p$, Bob pošle Alici $\alpha^y \bmod p$. Alice pak provede $(\alpha^y)^x \bmod p$, Bob obdobně.

Hodnota α se v praxi volí 2 nebo 5.

Výměna klíčů je založena na faktu, že v modulární aritmetice se velmi těžko hledá **diskrétní logaritmus** celého čísla, tedy číslo $x = \log_g(h) \in \mathbb{Z}/n\mathbb{Z}^*$ takové, že $g^x \equiv h \pmod n$.

Příklad: Uvažujme kongruenci $3^x \equiv 15 \pmod{19}$. Jedním z možných řešení je $x = 5$, neboť $3^5 \equiv 15 \pmod{19}$, není to ale řešení jediné. Z Malé Fermatovy věty totiž plyne $3^{18} \equiv 1 \pmod{19}$ a proto $3^{5+18k} \equiv 15 \pmod{19}$ pro libovolné $k \in \mathbb{N}$. Zadanou kongruenci tedy splňují taková x, pro něž platí $x \equiv 5 \pmod{18}$ a ze zveřejněného $3^x \bmod 19$ jenom velkou náhodou určíme ono konkrétní x, zvolené Alicí.

Diffieho-Hellmanova výměna klíčů Vysvětlení

Vzhledem k tomu, že $[a]_p \cdot [b]_p = [ab]_p$ platí pro Alicí přijaté Bobovo $\alpha^y \bmod p$ následující:

$$\alpha^y \bmod p \equiv [\underbrace{\alpha \cdot \alpha \cdots \alpha}_{y-\text{krát}}]_p,$$

a po umocnění na x-tou:

$$\left([\underbrace{\alpha \cdot \alpha \cdots \alpha}_{y-\text{krát}}]_p\right)^x = \underbrace{[\underbrace{\alpha \cdot \alpha \cdots \alpha}_{y-\text{krát}}]_p \cdot [\underbrace{\alpha \cdot \alpha \cdots \alpha}_{y-\text{krát}}]_p \cdots [\underbrace{\alpha \cdot \alpha \cdots \alpha}_{y-\text{krát}}]_p}_{x-\text{krát}} \equiv$$

$$\equiv [\underbrace{\alpha \cdot \alpha \cdots \alpha}_{xy-\text{krát}}]_p.$$

Recipročně to platí i pro Bobem přijaté Aličino $\alpha^x \bmod p$.

Diffieho-Hellmanova výměna klíčů Příklad

Příklad výměny pro $p = 17$ a $\alpha = 5$

Alice si zvolí $x = 1039$.

Bob si zvolí $y = 1271$.

Po nezašifrovaném spojení pošle Alice Bobovi $5^{1039} \bmod 17 = 7$ a Bob pošle Alici $5^{1271} \bmod 17 = 10$.

Bob si spočte svůj klíč jako $7^{1271} \bmod 17 = 12$, Alice jako $10^{1039} \bmod 17 = 12$.

Ve skutečnosti budou p, α, x, y mnohem větší čísla (proč)?

Pro ilustraci situace útočníka si povšimněte, že platí $5^7 \equiv 5^{23} \equiv 5^{39} \equiv 5^{7+k\cdot 16} \equiv 10 \pmod{17}$ pro libovolné $k \in \mathbb{N}$ a že $1271 = 7 + 79 \cdot 16$. Stejně tak je $5^{15} \equiv 5^{31} \equiv 5^{47} \equiv 5^{15+k\cdot 16} \equiv 7 \pmod{17}$ a $1039 = 15 + 64 \cdot 16$. V našem ilustračním případě může útočník celkem lehce vyzkoušet všechny možné kombinace klíčů (bude jich jenom sedmnáct), ale v případě velkých prvočísel to už nebude praktické: Bude-li $p = 429183283$ a dokážeme-li otestovat 100 klíčů za vteřinu, bude prohledávání celého prostoru možných klíčů trvat přibližně 1192 hodin. Pro p z oblasti 64bitových čísel by prohledávání celého prostoru možných klíčů rychlostí 10^6 klíčů/s mohlo trvat až 585 tisíc let.

2.3 Modulární mocnění

Modulární mocnění Výpočet $c \equiv b^r \pmod{n}$

Neefektivně lze opakovaným násobením a redukcí:

$$c = \underbrace{b\,[\,b\,[\ldots[\,b \cdot b \mod n\,]\ldots]\,\mathrm{mod}\,n\,]\,\mathrm{mod}\,n}_{r-\text{krát}}$$

Opakovaný kvadrát

Efektivní algoritmus pro $b, r \in \mathbb{N}$ je následující

1. Nechť $r = \sum_{j=0}^{k} a_j \cdot 2^j, a_j \in \{0, 1\}$
2. Inicializujeme $c = 1 + a_0 \cdot (b - 1)$ a $b_0 = b$

3. Opakujeme pro $j = 1 \ldots k$:
   (a) Spočteme $b_j = b_{j-1}^2 \bmod n$
   (b) Pokud je $a_j > 0$, přepíšeme $c \leftarrow c \cdot b_j \bmod n$
4. Výsledkem je $c \equiv b^r \pmod n$

Je možná jednodušší si postup přiblížit následujícím příkladem: Mám-li počítat $a \equiv 21^{41} \pmod{43}$, nebudu postupně vyčíslovat $a_1 = 21 \cdot 21 \bmod 43$, $a_2 = a_1 \cdot 21 \bmod 43$ až $a = a_3 \cdot 21 \bmod 43$. Jednodušší je si uvědomit, že $41 = 32 + 8 + 1$ a že tedy $a \equiv 21^41 \pmod{43} \equiv 21 \cdot 21^8 \cdot 21^{32} \pmod{43}$, kde pro výpočet dvojkových mocnin čísla 21 potřebujeme pouze opakovaně umocňovat na druhou:

$$\begin{aligned} 21^2 &\equiv 11 \pmod{43}, \\ 21^4 &\equiv 11^2 \pmod{43} \equiv 35 \pmod{43}, \\ 21^8 &\equiv 35^2 \pmod{43} \equiv 21 \pmod{43}, \\ 21^{16} &\equiv 21^2 \pmod{43} \equiv 11 \pmod{43}, \\ 21^{32} &\equiv 11^2 \pmod{43} \equiv 35 \pmod{43}. \end{aligned}$$

Výsledek obdržíme jako $a \equiv 21 \cdot 21^8 \cdot 21^{32} \pmod{43} \equiv 21 \cdot 21 \cdot 35 \pmod{43}$ a po úpravách $a \equiv 41 \pmod{43}$.

Počet kroků nutných pro umocnění b^r opakovaným kvadrátem je $\log_2 r$ oproti r krokům potřebným pro opakované násobení.

Modulární mocnění Příklad

Spočtěte $c = 3^{17} \bmod 7$

Nejprve rozložíme $r = 17 = 10001_\text{b}$. Je $a_0 = 1$ a proto prvotní hodnota $c = b = 3$ a $b_0 = 3$. Potom

$$\begin{aligned} b_1 &= 3^2 \bmod 7 = 9 \bmod 7 = 2, a_1 = 0, \\ b_2 &= 2^2 \bmod 7 = 4 \bmod 7 = 4, a_2 = 0, \\ b_3 &= 4^2 \bmod 7 = 16 \bmod 7 = 2, a_3 = 0, \\ b_4 &= 2^2 \bmod 7 = 4 \bmod 7 = 4, a_4 = 1. \end{aligned}$$

Nyní přepočteme $c = 3 \cdot 4 \bmod 7 = 12 \bmod 7 = 5$. Další binární cifry už v r nejsou, výsledkem je proto $c = 5$.

Kontrola: $3^{17} = 129140163 \bmod 7 = 5$.

2.4 RSA (Rivest, Shamir a Adelman 1977)

Šifrování veřejným klíčem Myšlenka RSA

RSA vychází z předpokladu, že faktorizace součinu prvočísel p a q je časově náročná – všichni proto mohou znát šifrovací klíč $n = p \cdot q$, ale nepomůže jim to ke zjištění dešifrovacího klíče, založeného na p a q.

V praxi je klíč $n = p \cdot q \in \{0,1\}^{1024}$ až $\{0,1\}^{4096}$.

Největší faktorizovaný RSA klíč je RSA-768 ($n = \{0,1\}^{768}$, 232 dekadických číslic) v roce 2009 za 2,5 roku na síti několika set pracovních stanic.

Ale pozor: V květnu 2007 padlo $M_{1039} = 2^{1039} - 1$ za 11 měsíců v laboratořích EPFL, Uni Bonn a NTT.

Faktorizovat 1024-bitový RSA klíč by podle Kleinjunga a kolegů [2] bylo asi 1000× náročnější, než jejich faktorizace 768-bitového klíče. Vzhledem k tomu, že 768-bitový klíč je několikanásobně složitější na faktorizaci, než klíč o délce 512 bitů, a že mezi faktorizací těchto dvou klíčů uplynulo přibližně deset let, lze očekávat, že kilobitový klíč (tyto klíče se standardně používají v každodenním šifrovaném provozu) bude považován za faktorizovatelný (a tedy prolomitelný) už někdy během příštích deseti let.

Poslední faktorizovaný RSA klíč je přitom RSA-180, 596 bitů [1].

Algoritmy šifrování veřejným klíčem Prerekvizity

Většina algoritmů (a RSA rozhodně) staví na několika algebraických postupech:

- Modulární mocnění
- Modulární inverze $a^{-1} \cdot a \equiv 1 \pmod{n}$
- Čínská věta o zbytcích
- Eulerova funkce

Eulerova funkce $\phi(n)$ Rozšíření Malé Fermatovy věty

Leonhard Euler generalizoval Malou Fermatovu větu na

$$a^{\phi(n)} \equiv 1 \pmod{n},$$

kde $\phi(n)$ je již zmíněná **Eulerova funkce**:

- někdy se setkáte s názvem *totient*
- udává počet přirozených čísel $1 \leq x \leq n$, jež jsou s n nesoudělná
- pro prvočísla $\phi(p) = p - 1$

Pro nesoudělná x a y platí

$$\phi(x \cdot y) = \phi(x) \cdot \phi(y)$$

a pro prvočísla p, q také

$$\phi(p)\phi(q) = (p-1)(q-1).$$

Pro libovolné přirozené n platí také

$$\phi(n) = n \prod_{p|n} \left(1 - \frac{1}{p}\right).$$

Algoritmus RSA Generování veřejného a soukromého klíče

Přípravná fáze:

1. Zvolíme nepříliš si blízká prvočísla p a q.
2. Spočteme **modul** šifrovací a dešifrovací transformace, $n = p \cdot q$.
3. Vypočteme Eulerovu funkci pro n, $\phi(n) = (p-1)(q-1)$.
4. Zvolíme **šifrovací exponent** e takový, že $1 < e < \phi(n)$ a $\gcd(e, n) = 1$.
5. Dopočteme **dešifrovací exponent** d tak, aby d bylo multiplikativní inverzí k e modulo $\phi(n)$, $d \cdot e \equiv 1 \pmod{\phi(n)}$.

Veřejný klíč pro zašifrování zprávy je (n, e), **soukromý klíč** pro dešifrování je (n, d).

Algoritmus RSA Jak to funguje

Princip přenosu zprávy X je primitivní:

Šifrování

Po lince přenášíme šifrovaný text c, jenž vznikne jako

$$c = X^e \bmod n.$$

Dešifrování

Příjemce si z přijatého šifrovaného textu spočítá původní zprávu jako

$$X = c^d \bmod n.$$

Trik celého postupu spočívá v tom, že *z pouhé znalosti* (n, e) *nelze v rozumném čase určit* d.

Obdržíme dešifrováním opravdu původní text?

Při dešifrování $c \equiv X^e \pmod n$ máme

$$c^d \equiv (X^e)^d \equiv X^{ed} \pmod n \equiv X^{ed} \pmod{pq},$$

a $\gcd(p, q) = 1$, což připomíná řešení CRT o dvou kongruencích.

Prozkoumáme vlastnosti $c^d \equiv X^{ed} \pmod p$ a $c^d \equiv X^{ed} \pmod q$ a zobecníme je na operace modulo n.

Z definice součinu ed v algoritmu RSA plyne

$$ed \equiv 1 \pmod{\phi(n)} \Rightarrow \exists f \in \mathbb{Z} : ed = 1 + f(p-1)(q-1),$$

což můžeme dále upravit na

$$ed = 1 + f(p-1)(q-1) = 1 + g(q-1) = 1 + h(p-1)$$

a tedy

$$ed \equiv 1 \pmod{\phi(n)} \equiv 1 \pmod{\phi(q)} \equiv 1 \pmod{\phi(p)}.$$

Důkaz: $ed \equiv 1 \pmod{\phi(n)} \equiv 1 \pmod{(p-1)(q-1)} \Leftrightarrow ed = 1 + g(p-1)(q-1)$ a tedy $ed = 1 + g_p(q-1)$ a $ed = 1 + g_q(p-1)$.

Dokazujeme stále pro p a q odděleně:

Pro $p \nmid X$ je podle Malé Fermatovy věty $X^{p-1} \equiv 1 \pmod p$ a tedy

$$X^{ed} = X^{1+h(p-1)} = X^{h(p-1)}X = \left(X^{(p-1)}\right)^h X \equiv 1^h X \equiv X \pmod p.$$

Pro $p|X$ je

$$X^{ed} \equiv 0^{ed} \pmod p \equiv X \pmod p$$

To samé platí pro q a tedy

$$X^{ed} \equiv X \pmod p$$
$$X^{ed} \equiv X \pmod q$$

Jedním z důsledků CRT je pro nesoudělná x a y ekvivalence

$$\begin{array}{rcl} a & \equiv & b \pmod{x} \\ a & \equiv & b \pmod{y} \end{array} \quad \Leftrightarrow \quad a \equiv b \pmod{xy}.$$

Proto také z

$$\begin{aligned} X^{ed} &\equiv X \pmod{p} \\ X^{ed} &\equiv X \pmod{q} \end{aligned}$$

plyne

$$X^{ed} \equiv X \pmod{pq}.$$

Ekvivalence

$$\begin{array}{rcl} a & \equiv & b \pmod{x} \\ a & \equiv & b \pmod{y} \end{array} \quad \Leftrightarrow \quad a \equiv b \pmod{xy}.$$

platí neboť z první rovnice

$$\begin{aligned} a &= k_1 x + b \\ a - b &= k_1 x \end{aligned}$$

a z druhé obdobně

$$\begin{aligned} a &= k_2 y + b \\ a - b &= k_2 y. \end{aligned}$$

Vzhledem k tomu, že x a y jsou nesoudělná, obě dělí výraz $a - b$,

$$a - b = k_1 x = k_2 y,$$

musí být

$$k_1 x = k_2 y = \kappa\, xy$$

a tedy také

$$a - b = \kappa\, xy$$

a proto

$$a \equiv b \pmod{xy}.$$

2.5 CRT-RSA

V úvodu přednášky zmíněná Čínská věta o zbytcích má velmi zajímavé využití právě při výpočtech dešifrovací části šifry RSA.

RSA pomocí CRT Urychlení dešifrování (1/3)

Jak modul n, tak i dešifrovací exponent d jsou hodně velká čísla, a proces dešifrování

$$X \equiv c^d \pmod{n}$$

trvá dlouho.

K urychlení dešifrovací transformace lze použít rozklad na výpočet s menšími moduly pomocí Čínské věty o zbytcích.

Pro $n = pq$ použijeme již jednou provedený trik

$$X \equiv X_p \pmod{p} \equiv c^{d_p} \pmod{p}$$
$$X \equiv X_q \pmod{q} \equiv c^{d_q} \pmod{q}$$

přičemž pro p

$$d_p \equiv d \pmod{\phi(p)} \Leftrightarrow d = d_p + j(p-1)$$

a tedy

$$c^d \equiv c^{d_p + j(p-1)} \pmod{p} \equiv c^{d_p} 1^j \pmod{p} \equiv c^{d_p} \pmod{p}$$

Pro q je to obdobně.

Plné znění je

$$d_p \equiv d \pmod{p-1} \Leftrightarrow d = d_p + j(p-1)$$
$$d_q \equiv d \pmod{q-1} \Leftrightarrow d = d_q + k(q-1)$$

a tedy

$$c^d \equiv c^{d_p + j(p-1)} \pmod{p} \equiv c^{d_p} 1^j \pmod{p} \equiv c^{d_p} \pmod{p}$$
$$c^d \equiv c^{d_q + k(q-1)} \pmod{q} \equiv c^{d_q} 1^k \pmod{q} \equiv c^{d_q} \pmod{q}$$

Zpráva X je tedy řešením soustavy dvou kongruencí sestavených pro c:

$$X \equiv c^{d_p} \pmod{p},$$
$$X \equiv c^{d_q} \pmod{q}.$$

Řešením je

$$X = \left[c^{d_p} M_p q + c^{d_q} M_q p\right] \bmod pq,$$

kde $M_p = q^{-1} \bmod p$ a $M_q = p^{-1} \bmod q$.

Prolomení RSA při nevhodné volbě p a q Prolomení při nevhodné volbě p a q

Pokud zvolím p a q nevhodně (blízko sebe, příliš malá, atd.), útočník využije znalosti (n, e):

1. Faktorizuje n na p a q.
2. Vypočte Eulerovu funkci pro n, $\phi(n) = (p-1)(q-1)$.
3. Dopočte **dešifrovací exponent** d tak, aby $d \cdot e \equiv 1 \pmod{\phi(n)}$.

Můj **soukromý klíč** pro dešifrování (n, d) v ten okamžik zná i útočník a může moje zprávy dešifrovat.

Příklad RSA: doplnit, externí viz například `http://en.wikipedia.org/wiki/RSA#A_working_example`

3 Závěr

A co nás čeká příště?

Složitost algoritmů.

Reference

[1] Danilov, S. A. – Popovyan, I. A.: Factorization of RSA-180. [online] Cryptology ePrint Archive. Dostupné na WWW: `http://eprint.iacr.org/2010/270.pdf` (staženo 13.10.2010).

[2] Kleinjung, T. – Aoki, K. – Franke, J. – Lenstra, A. – Thome, E. – Bos, J. – Gaudry, P. – Kruppa, A. – Montgomery, P. – Osvik, D. A. – te Riele, H. – Timofeev, A. – Zimmermann, P. Factorization of a 768-bit RSA modulus. [online] Cryptology ePrint Archive. Dostupné na WWW: `http://eprint.iacr.org/2010/006.pdf` (staženo 13.10.2010).