CIMPA School on Number Theory and Cryptography

L'ubomíra Balková a Štěpán Starosta

ACS seminář

23. listopadu 2010

Věnováno Prof. Karlu Kloudovi, **Ph.D.**

Program CIMPA School on Number Theory and Cryptography

1. Kalyan Chakraborty (Harish Chandra Research Institute) : Introduction to Basic Cryptography
2. Roberto Ferretti (Roma Tre) : Selected Numerical Methods
3. Corrado Falcolini (Roma Tre) : Wolfram Mathematica Laboratory
4. Francesco Pappalardi (Roma Tre): Algorithmic Number Theory
5. Michel Waldschmidt (Paris) : Finite Fields
6. Pierre Arnoux (Luminy) : Primality Testing and Factoring Algorithms
7. Shigeru Kanemitsu (Fukuoka) : The Riemann Zeta Function
8. Christian Mauduit (Luminy) : Random Sequences and Cryptography
9. Jorge Jimenez Urroz (Universitat Politecnica de Catalunya) : Elliptic Curves in Cryptography
10. Kanhaiya Jha (KU, Nepal) : Division Algorithm and Fixed Point

Program CIMPA School on Number Theory and Cryptography

1. **Kalyan Chakraborty (Harish Chandra Research Institute) : Introduction to Basic Cryptography**
2. Roberto Ferretti (Roma Tre) : Selected Numerical Methods
3. **Corrado Falcolini (Roma Tre) : Wolfram Mathematica Laboratory**
4. Francesco Pappalardi (Roma Tre): Algorithmic Number Theory
5. Michel Waldschmidt (Paris) : Finite Fields
6. **Pierre Arnoux (Luminy) : Primality Testing and Factoring Algorithms**
7. Shigeru Kanemitsu (Fukuoka) : The Riemann Zeta Function
8. Christian Mauduit (Luminy) : Random Sequences and Cryptography
9. Jorge Jimenez Urroz (Universitat Politecnica de Catalunya) : Elliptic Curves in Cryptography
10. Kanhaiya Jha (KU, Nepal) : Division Algorithm and Fixed Point

Opáčko z algebry

$m > 1$

aditivní grupa $(\mathbb{Z}/m\mathbb{Z}, +)$

multiplikativní grupa $((\mathbb{Z}/m\mathbb{Z})^*, \times)$

Opáčko z algebry

$m > 1$
aditivní grupa $(\mathbb{Z}/m\mathbb{Z}, +)$

multiplikativní grupa $((\mathbb{Z}/m\mathbb{Z})^*, \times)$

řád prvku, řád grupy

Opáčko z algebry

$m > 1$
aditivní grupa $(\mathbb{Z}/m\mathbb{Z}, +)$

multiplikativní grupa $((\mathbb{Z}/m\mathbb{Z})^*, \times)$

řád prvku, řád grupy

cyklická grupa, generátor g: $< g >= G$

Opáčko z algebry

$m > 1$
aditivní grupa $(\mathbb{Z}/m\mathbb{Z}, +)$

multiplikativní grupa $((\mathbb{Z}/m\mathbb{Z})^*, \times)$

řád prvku, řád grupy

cyklická grupa, generátor g: $< g >= G$

multiplikativní grupa je cyklická $\Leftrightarrow$ m je 2, 4, p^k nebo $2p^k$, kde p je liché prvočíslo

Opáčko z algebry

$m > 1$
aditivní grupa $(\mathbb{Z}/m\mathbb{Z}, +)$

multiplikativní grupa $((\mathbb{Z}/m\mathbb{Z})^*, \times)$

řád prvku, řád grupy

cyklická grupa, generátor g: $< g >= G$

multiplikativní grupa je cyklická $\Leftrightarrow$ m je 2, 4, p^k nebo $2p^k$, kde p je liché prvočíslo

Modulární aritmetika

Rychlost operací v okruhu $\mathbb{Z}/m\mathbb{Z}$

	jednoduše	sofistikovaně
sčítání	$\mathcal{O}(\log m)$	
násobení	$\mathcal{O}(\log^2 m)$	$\mathcal{O}(\log^{1+\varepsilon} m)$
mocnění na n	$\mathcal{O}(\log n \log^2 m)$	$\mathcal{O}(\log n \log^{1+\varepsilon} m)$

NSD lze též spočítat relativně rychle

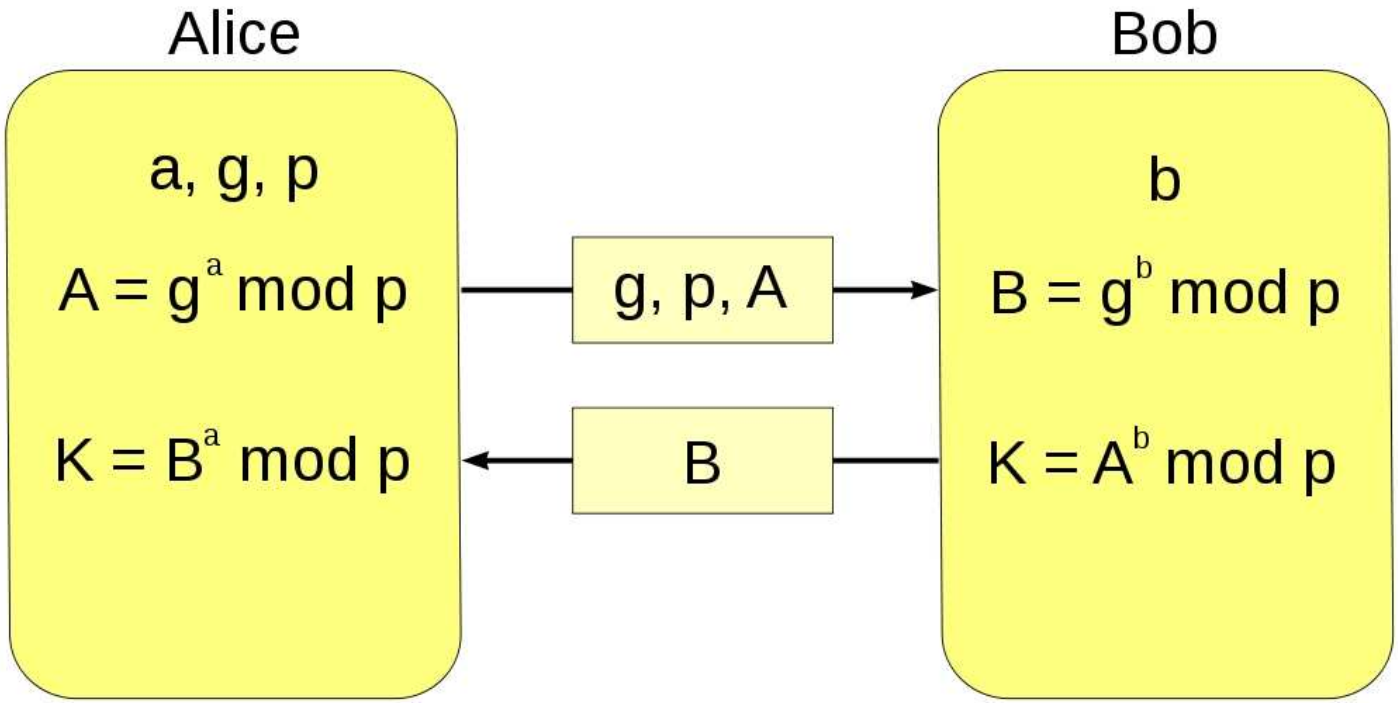
Modulární aritmetika

Rychlost operací v okruhu $\mathbb{Z}/m\mathbb{Z}$

	jednoduše	sofistikovaně
sčítání	$\mathcal{O}(\log m)$	
násobení	$\mathcal{O}(\log^2 m)$	$\mathcal{O}(\log^{1+\varepsilon} m)$
mocnění na n	$\mathcal{O}(\log n \log^2 m)$	$\mathcal{O}(\log n \log^{1+\varepsilon} m)$

NSD lze též spočítat relativně rychle

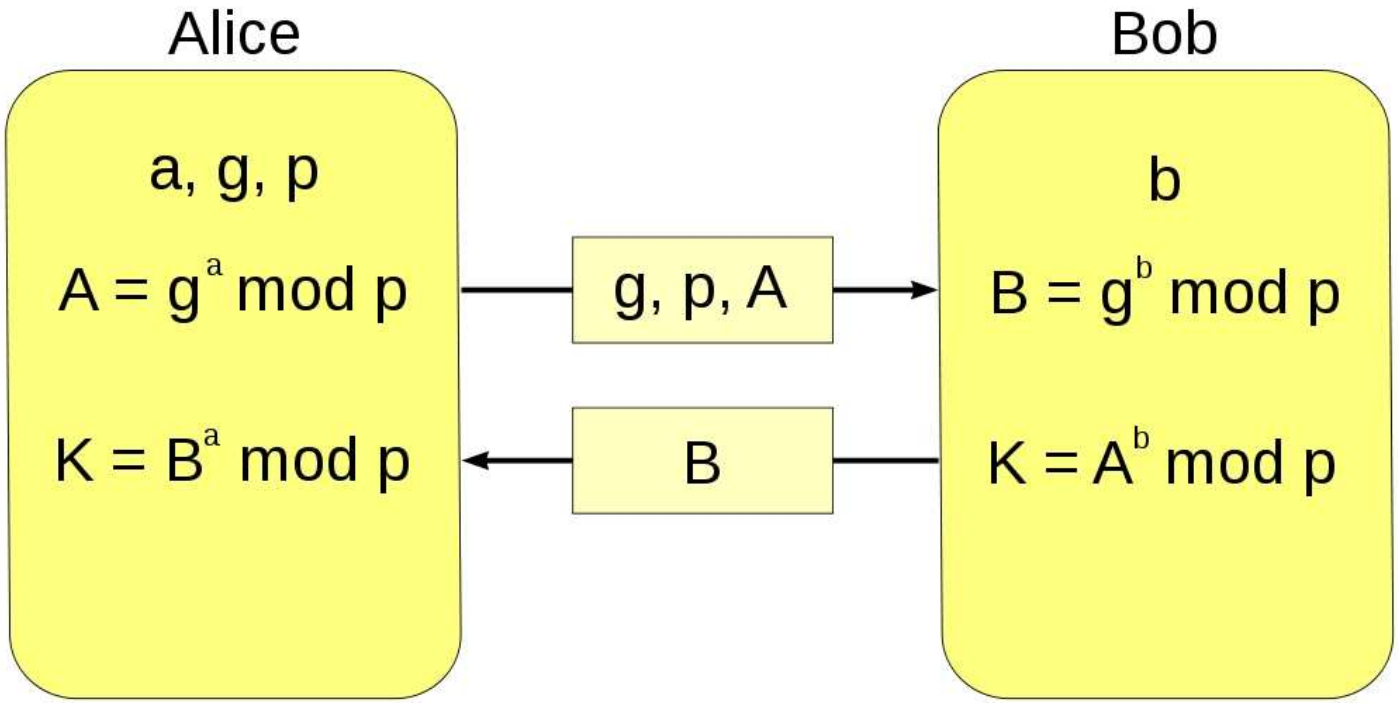
Diffie-Hellman key exchange a DLOG



p prvočíslo, g generátor $(\mathbb{Z}/p\mathbb{Z})^*$, $a, b \in (\mathbb{Z}/p\mathbb{Z})^*$

nepřítel nemá a a b a potřeboval vyřešit **diskrétní logaritmus**
(na stejném principu je například založen i ElGamal signature scheme)

Diffie-Hellman key exchange a DLOG



p prvočíslo, g generátor $(\mathbb{Z}/p\mathbb{Z})^*$, $a, b \in (\mathbb{Z}/p\mathbb{Z})^*$

nepřítel nemá a a b a potřeboval vyřešit **diskrétní logaritmus**
(na stejném principu je například založen i ElGamal signature scheme)

Testování prvočíselnosti a faktorizace

1. **Primality problem**: Rozhodni, zda je dané $n \in \mathbb{N}$ prvočíslo.
2. **Factoring problem**: Najdi prvočíselný rozklad daného $n \in \mathbb{N}$.

- po staletí jen teoretický význam

Testování prvočíselnosti a faktorizace

1. **Primality problem**: Rozhodni, zda je dané $n \in \mathbb{N}$ prvočíslo.
2. **Factoring problem**: Najdi prvočíselný rozklad daného $n \in \mathbb{N}$.

- po staletí jen teoretický význam
- v současnosti obrovský praktický význam - díky *kryptologii*

Testování prvočíselnosti a faktorizace

1. **Primality problem**: Rozhodni, zda je dané $n \in \mathbb{N}$ prvočíslo.
2. **Factoring problem**: Najdi prvočíselný rozklad daného $n \in \mathbb{N}$.

- po staletí jen teoretický význam
- v současnosti obrovský praktický význam - díky *kryptologii*

Algoritmus RSA - Rivest, Shamir, Adleman 1978

Definice: $\varphi(n) =$ počet čísel $\in \{1, 2, \ldots, n-1\}$ nesoudělných s n

POZN.: $\varphi(p \cdot q) = (p-1)(q-1)$ pro $p, q \in \mathbb{P}$

Algoritmus RSA - Rivest, Shamir, Adleman 1978

Definice: $\varphi(n) =$ počet čísel $\in \{1, 2, \ldots, n-1\}$ nesoudělných s n

POZN.: $\varphi(p \cdot q) = (p-1)(q-1)$ pro $p, q \in \mathbb{P}$

Věta (Eulerova-Fermatova): Nechť $m, n \in \mathbb{N}$.

Pak $NSD(m, n) = 1 \Leftrightarrow m^{\varphi(n)} \bmod n \equiv 1$.

Algoritmus RSA - Rivest, Shamir, Adleman 1978

Definice: $\varphi(n) =$ počet čísel $\in \{1, 2, \ldots, n-1\}$ nesoudělných s n
POZN.: $\varphi(p \cdot q) = (p-1)(q-1)$ pro $p, q \in \mathbb{P}$
Věta (Eulerova-Fermatova): Nechť $m, n \in \mathbb{N}$.
Pak $NSD(m, n) = 1 \Leftrightarrow m^{\varphi(n)} \bmod n \equiv 1$.

1. *generování klíče* **Primality problem**
   - Bob generuje velká $p, q \in \mathbb{P}$, $n := pq$ ... *modul RSA*
   - zvolí $1 < e < \varphi(n) = (p-1)(q-1)$ a $\mathrm{NSD}(e, \varphi(n)) = 1$
   - Eukleidovým algoritmem najde jediné $1 < d < \varphi(n)$... *soukromý klíč*

     $$ed \bmod \varphi(n) \equiv 1, \quad \text{tj. } ed = \varphi(n)k + 1 \text{ pro } k \in \mathbb{N}$$

   - zveřejní (n, e) ... *veřejný klíč*

Algoritmus RSA - Rivest, Shamir, Adleman 1978

Definice: $\varphi(n) =$ počet čísel $\in \{1, 2, \ldots, n-1\}$ nesoudělných s n
POZN.: $\varphi(p \cdot q) = (p-1)(q-1)$ pro $p, q \in \mathbb{P}$
Věta (Eulerova-Fermatova): Nechť $m, n \in \mathbb{N}$.
Pak $NSD(m, n) = 1 \Leftrightarrow m^{\varphi(n)} \bmod n \equiv 1$.

1. *generování klíče* **Primality problem**
   - Bob generuje velká $p, q \in \mathbb{P}$, $n := pq$ ... *modul RSA*
   - zvolí $1 < e < \varphi(n) = (p-1)(q-1)$ a $\mathrm{NSD}(e, \varphi(n)) = 1$
   - Eukleidovým algoritmem najde jediné $1 < d < \varphi(n)$... *soukromý klíč*

$$ed \bmod \varphi(n) \equiv 1, \quad \text{tj. } ed = \varphi(n)k + 1 \text{ pro } k \in \mathbb{N}$$

   - zveřejní (n, e) ... *veřejný klíč*
2. *šifrování*
   - Alice vyhledá v databázi Bobův klíč (n, e)
   - zprávu $m < n$ šifruje $c \equiv m^e \bmod n$

Algoritmus RSA - Rivest, Shamir, Adleman 1978

Definice: $\varphi(n) =$ počet čísel $\in \{1, 2, \ldots, n-1\}$ nesoudělných s n
POZN.: $\varphi(p \cdot q) = (p-1)(q-1)$ pro $p, q \in \mathbb{P}$
Věta (Eulerova-Fermatova): Nechť $m, n \in \mathbb{N}$.
Pak $NSD(m, n) = 1 \Leftrightarrow m^{\varphi(n)} \bmod n \equiv 1$.

1. *generování klíče* **Primality problem**
   - Bob generuje velká $p, q \in \mathbb{P}$, $n := pq$ ... *modul RSA*
   - zvolí $1 < e < \varphi(n) = (p-1)(q-1)$ a $\mathrm{NSD}(e, \varphi(n)) = 1$
   - Eukleidovým algoritmem najde jediné $1 < d < \varphi(n)$... *soukromý klíč*

$$ed \bmod \varphi(n) \equiv 1, \quad \text{tj. } ed = \varphi(n)k + 1 \text{ pro } k \in \mathbb{N}$$

   - zveřejní (n, e) ... *veřejný klíč*
2. *šifrování*
   - Alice vyhledá v databázi Bobův klíč (n, e)
   - zprávu $m < n$ šifruje $c \equiv m^e \bmod n$
3. *dešifrování*
   - Bob spočte $m \equiv c^d \bmod n$

Algoritmus RSA - Rivest, Shamir, Adleman 1978

Definice: $\varphi(n) =$ počet čísel $\in \{1, 2, \ldots, n-1\}$ nesoudělných s n
POZN.: $\varphi(p \cdot q) = (p-1)(q-1)$ pro $p, q \in \mathbb{P}$
Věta (Eulerova-Fermatova): Nechť $m, n \in \mathbb{N}$.
Pak $NSD(m, n) = 1 \Leftrightarrow m^{\varphi(n)} \bmod n \equiv 1$.

1. *generování klíče* **Primality problem**
   - Bob generuje velká $p, q \in \mathbb{P}$, $n := pq$ ... *modul RSA*
   - zvolí $1 < e < \varphi(n) = (p-1)(q-1)$ a $\mathrm{NSD}(e, \varphi(n)) = 1$
   - Eukleidovým algoritmem najde jediné $1 < d < \varphi(n)$... *soukromý klíč*

     $$ed \bmod \varphi(n) \equiv 1, \quad \text{tj. } ed = \varphi(n)k + 1 \text{ pro } k \in \mathbb{N}$$

   - zveřejní (n, e) ... *veřejný klíč*
2. *šifrování*
   - Alice vyhledá v databázi Bobův klíč (n, e)
   - zprávu $m < n$ šifruje $c \equiv m^e \bmod n$
3. *dešifrování*
   - Bob spočte $m \equiv c^d \bmod n$

Algoritmus RSA - Rivest, Shamir, Adleman 1978

Věta (Eulerova-Fermatova): Nechť $m, n \in \mathbb{N}$.
Pak $NSD(m, n) = 1 \Leftrightarrow m^{\varphi(n)} \bmod n \equiv 1$.

- *dešifrování*
  - Bob spočte $c^d \bmod n$

$$\begin{aligned} c^d \bmod n &\equiv (m^e)^d \bmod n \\ &\equiv m^{ed} \bmod n \\ &\equiv m^{\varphi(n)k+1} \bmod n \\ &\equiv m \cdot (m^{\varphi(n)})^k \bmod n \\ &= m \end{aligned}$$

Algoritmus RSA - Rivest, Shamir, Adleman 1978

Věta (Eulerova-Fermatova): Nechť $m, n \in \mathbb{N}$.
Pak $NSD(m, n) = 1 \Leftrightarrow m^{\varphi(n)} \bmod n \equiv 1$.

- *dešifrování*
  - Bob spočte $c^d \bmod n$

$$\begin{aligned} c^d \bmod n &\equiv (m^e)^d \bmod n \\ &\equiv m^{ed} \bmod n \\ &\equiv m^{\varphi(n)k+1} \bmod n \\ &\equiv m \cdot (m^{\varphi(n)})^k \bmod n \\ &= m \end{aligned}$$

- *kryptoanalýza* **Factoring problem**
  - rozklad $n = pq$

Algoritmus RSA - Rivest, Shamir, Adleman 1978

Věta (Eulerova-Fermatova): Nechť $m, n \in \mathbb{N}$.
Pak $NSD(m, n) = 1 \Leftrightarrow m^{\varphi(n)} \bmod n \equiv 1$.

- *dešifrování*
  - Bob spočte $c^d \bmod n$

$$\begin{aligned} c^d \bmod n &\equiv (m^e)^d \bmod n \\ &\equiv m^{ed} \bmod n \\ &\equiv m^{\varphi(n)k+1} \bmod n \\ &\equiv m \cdot (m^{\varphi(n)})^k \bmod n \\ &= m \end{aligned}$$

- *kryptoanalýza* **Factoring problem**
  - rozklad $n = pq$

Pravděpodobnostní test

1. **Primality Testing**
2. Factorization

Rabin-Millerův test

pravděpodobnostní - rychlý, složitost $\mathcal{O}^{\sim}(\log^3 n) = \mathcal{O}(\log^3 n P(\log\log n))$

obecně: $\mathcal{O}^{\sim}(f) = \mathcal{O}(fP(\log f))$

Rabin-Millerův test

pravděpodobnostní - rychlý, složitost $\mathcal{O}^{\sim}(\log^3 n) = \mathcal{O}(\log^3 n P(\log\log n))$
obecně: $\mathcal{O}^{\sim}(f) = \mathcal{O}(fP(\log f)\ )$

- n liché prvočíslo a $n - 1 = 2^k t$

Rabin-Millerův test

pravděpodobnostní - rychlý, složitost $\mathcal{O}^{\sim}(\log^3 n) = \mathcal{O}(\log^3 n P(\log\log n))$
obecně: $\mathcal{O}^{\sim}(f) = \mathcal{O}(fP(\log f))$

- n liché prvočíslo a $n - 1 = 2^k t$
- Malá Fermatova věta $\Rightarrow$ pro každé $a < n$ platí

$$0 \equiv (a^{n-1} - 1) \mod n \equiv (a^{2^k t} - 1) \mod n$$

Rabin-Millerův test

pravděpodobnostní - rychlý, složitost $\mathcal{O}^{\sim}(\log^3 n) = \mathcal{O}(\log^3 n P(\log\log n))$
obecně: $\mathcal{O}^{\sim}(f) = \mathcal{O}(fP(\log f))$

- n liché prvočíslo a $n - 1 = 2^k t$
- Malá Fermatova věta $\Rightarrow$ pro každé $a < n$ platí

$$0 \equiv (a^{n-1} - 1) \mod n \equiv (a^{2^k t} - 1) \mod n$$

- n nutně dělí aspoň jednu ze závorek:

$$(a^{2^{k-1}t} - 1)(a^{2^{k-1}t} + 1) = (a^{2^{k-2}t} - 1)(a^{2^{k-2}t} + 1)(a^{2^{k-1}t} + 1) =$$

Rabin-Millerův test

pravděpodobnostní - rychlý, složitost $\mathcal{O}^{\sim}(\log^3 n) = \mathcal{O}(\log^3 n P(\log\log n))$
obecně: $\mathcal{O}^{\sim}(f) = \mathcal{O}(fP(\log f)\ )$

- n liché prvočíslo a $n-1 = 2^k t$
- Malá Fermatova věta $\Rightarrow$ pro každé $a < n$ platí

$$0 \equiv (a^{n-1} - 1) \mod n \equiv (a^{2^k t} - 1) \mod n$$

- n nutně dělí aspoň jednu ze závorek:

$$(a^{2^{k-1}t} - 1)(a^{2^{k-1}t} + 1) = (a^{2^{k-2}t} - 1)(a^{2^{k-2}t} + 1)(a^{2^{k-1}t} + 1) =$$

$$= \underline{(a^t - 1)(a^t + 1)\ldots(a^{2^{k-2}t} + 1)(a^{2^{k-1}t} + 1).}$$

Rabin-Millerův test

pravděpodobnostní - rychlý, složitost $\mathcal{O}^{\sim}(\log^3 n) = \mathcal{O}(\log^3 n P(\log\log n))$
obecně: $\mathcal{O}^{\sim}(f) = \mathcal{O}(fP(\log f))$

- n liché prvočíslo a $n-1 = 2^k t$
- Malá Fermatova věta $\Rightarrow$ pro každé $a < n$ platí

$$0 \equiv (a^{n-1} - 1) \mod n \equiv (a^{2^k t} - 1) \mod n$$

- n nutně dělí aspoň jednu ze závorek:

$$(a^{2^{k-1}t} - 1)(a^{2^{k-1}t} + 1) = (a^{2^{k-2}t} - 1)(a^{2^{k-2}t} + 1)(a^{2^{k-1}t} + 1) =$$

$$= \underline{(a^t - 1)(a^t + 1)\ldots(a^{2^{k-2}t} + 1)(a^{2^{k-1}t} + 1).}$$

- n složené číslo, pak existuje $a < n$, pro které n nedělí žádnou ze závorek

Rabin-Millerův test

pravděpodobnostní - rychlý, složitost $\mathcal{O}^{\sim}(\log^3 n) = \mathcal{O}(\log^3 n P(\log\log n))$
obecně: $\mathcal{O}^{\sim}(f) = \mathcal{O}(fP(\log f))$

- n liché prvočíslo a $n-1 = 2^k t$
- Malá Fermatova věta $\Rightarrow$ pro každé $a < n$ platí

$$0 \equiv (a^{n-1} - 1) \mod n \equiv (a^{2^k t} - 1) \mod n$$

- n nutně dělí aspoň jednu ze závorek:

$$(a^{2^{k-1}t} - 1)(a^{2^{k-1}t} + 1) = (a^{2^{k-2}t} - 1)(a^{2^{k-2}t} + 1)(a^{2^{k-1}t} + 1) =$$

$$= \underline{(a^t - 1)(a^t + 1)\ldots(a^{2^{k-2}t} + 1)(a^{2^{k-1}t} + 1)}.$$

- n složené číslo, pak existuje $a < n$, pro které n nedělí žádnou ze závorek (takových a jsou aspoň 3/4, tj. $\frac{3(n-1)}{4}$)

Rabin-Millerův test

pravděpodobnostní - rychlý, složitost $\mathcal{O}^{\sim}(\log^3 n) = \mathcal{O}(\log^3 n P(\log\log n))$
obecně: $\mathcal{O}^{\sim}(f) = \mathcal{O}(f P(\log f))$

- n liché prvočíslo a $n-1 = 2^k t$
- Malá Fermatova věta $\Rightarrow$ pro každé $a < n$ platí

$$0 \equiv (a^{n-1} - 1) \mod n \equiv (a^{2^k t} - 1) \mod n$$

- n nutně dělí aspoň jednu ze závorek:

$$(a^{2^{k-1}t} - 1)(a^{2^{k-1}t} + 1) = (a^{2^{k-2}t} - 1)(a^{2^{k-2}t} + 1)(a^{2^{k-1}t} + 1) =$$

$$= \underline{(a^t - 1)(a^t + 1)\ldots(a^{2^{k-2}t} + 1)(a^{2^{k-1}t} + 1)}.$$

- n složené číslo, pak existuje $a < n$, pro které n nedělí žádnou ze závorek (takových a jsou aspoň 3/4, tj. $\frac{3(n-1)}{4}$)

Algoritmus Rabin-Millerova testu

- testujeme, zda n je prvočíslo
- rozložíme $n - 1 = 2^k t$

Algoritmus Rabin-Millerova testu

- testujeme, zda n je prvočíslo
- rozložíme $n - 1 = 2^k t$
- bereme libovolná $a_1, a_2, \ldots, a_r \in \{1, 2, \ldots, n-1\}$

Algoritmus Rabin-Millerova testu

- testujeme, zda n je prvočíslo
- rozložíme $n - 1 = 2^k t$
- bereme libovolná $a_1, a_2, \ldots, a_r \in \{1, 2, \ldots, n-1\}$
- kontrolujeme $NSD(a_i, n) = 1$

Algoritmus Rabin-Millerova testu

- testujeme, zda n je prvočíslo
- rozložíme $n - 1 = 2^k t$
- bereme libovolná $a_1, a_2, \ldots, a_r \in \{1, 2, \ldots, n-1\}$
- kontrolujeme $NSD(a_i, n) = 1$
- pokud pro některé a_i platí:
  - $a_i^t \bmod n \not\equiv \pm 1$,
  - $a_i^{t2^j} \bmod n \not\equiv -1$ pro každé $j \in \{1, \ldots, k-1\}$,

$\Rightarrow$ n je složené

Algoritmus Rabin-Millerova testu

- testujeme, zda n je prvočíslo
- rozložíme $n - 1 = 2^k t$
- bereme libovolná $a_1, a_2, \ldots, a_r \in \{1, 2, \ldots, n-1\}$
- kontrolujeme $NSD(a_i, n) = 1$
- pokud pro některé a_i platí:
  - $a_i^t \bmod n \not\equiv \pm 1$,
  - $a_i^{t2^j} \bmod n \not\equiv -1$ pro každé $j \in \{1, \ldots, k-1\}$,

  $\Rightarrow$ n je složené
- jinak je n prvočíslo s pravděpodobností $1 - \frac{1}{4^r}$

Algoritmus Rabin-Millerova testu

- testujeme, zda n je prvočíslo
- rozložíme $n - 1 = 2^k t$
- bereme libovolná $a_1, a_2, \ldots, a_r \in \{1, 2, \ldots, n-1\}$
- kontrolujeme $NSD(a_i, n) = 1$
- pokud pro některé a_i platí:
  - $a_i^t \bmod n \not\equiv \pm 1$,
  - $a_i^{t2^j} \bmod n \not\equiv -1$ pro každé $j \in \{1, \ldots, k-1\}$,

  $\Rightarrow$ n je složené
- jinak je n prvočíslo s pravděpodobností $1 - \frac{1}{4^r}$

Příklad: Rabin-Millerův test

- testujeme, zda 49 je prvočíslo
- rozložíme $49 - 1 = 2^4 \cdot 3 = 2^k t$

Příklad: Rabin-Millerův test

- testujeme, zda 49 je prvočíslo
- rozložíme $49 - 1 = 2^4 \cdot 3 = 2^k t$
- $a_1 := 2$

Příklad: Rabin-Millerův test

- testujeme, zda 49 je prvočíslo
- rozložíme $49 - 1 = 2^4 \cdot 3 = 2^k t$
- $a_1 := 2$
- kontrolujeme $NSD(2, 49) = 1$

Příklad: Rabin-Millerův test

- testujeme, zda 49 je prvočíslo
- rozložíme $49 - 1 = 2^4 \cdot 3 = 2^k t$
- $a_1 := 2$
- kontrolujeme $NSD(2, 49) = 1$
- dělí 49 některou závorku?
  $(2^3 - 1)(2^3 + 1)((2^3)^2 + 1)(((2^3)^2)^2 + 1)((((2^3)^2)^2)^2 + 1)$
  $\equiv 7 \cdot 9 \cdot 16 \cdot 30 \cdot 9 \bmod 49$

Příklad: Rabin-Millerův test

- testujeme, zda 49 je prvočíslo
- rozložíme $49 - 1 = 2^4 \cdot 3 = 2^k t$
- $a_1 := 2$
- kontrolujeme $NSD(2, 49) = 1$
- dělí 49 některou závorku?
  $(2^3 - 1)(2^3 + 1)((2^3)^2 + 1)(((2^3)^2)^2 + 1)((((2^3)^2)^2)^2 + 1)$
  $\equiv 7 \cdot 9 \cdot 16 \cdot 30 \cdot 9 \bmod 49$
- NE! $\Rightarrow$ 49 je složené

Příklad: Rabin-Millerův test

- testujeme, zda 49 je prvočíslo
- rozložíme $49 - 1 = 2^4 \cdot 3 = 2^k t$
- $a_1 := 2$
- kontrolujeme $NSD(2, 49) = 1$
- dělí 49 některou závorku?
  $(2^3 - 1)(2^3 + 1)((2^3)^2 + 1)(((2^3)^2)^2 + 1)((((2^3)^2)^2)^2 + 1)$
  $\equiv 7 \cdot 9 \cdot 16 \cdot 30 \cdot 9 \bmod 49$
- NE! $\Rightarrow$ 49 je složené

Logická otázka

"Kolik náhodných přirozených čísel je třeba otestovat, než najdeme prvočíslo?"

Definice: $\pi(n) =$ počet prvočísel $\leq n$

Logická otázka

"Kolik náhodných přirozených čísel je třeba otestovat, než najdeme prvočíslo?"

Definice: $\pi(n) =$ počet prvočísel $\leq n$

Prime Number Theorem: $\pi(n) \sim \frac{n}{\ln n}$

Logická otázka

"Kolik náhodných přirozených čísel je třeba otestovat, než najdeme prvočíslo?"

Definice: $\pi(n) =$ počet prvočísel $\leq n$

Prime Number Theorem: $\pi(n) \sim \frac{n}{\ln n}$

Důsledek: vybereme-li p náhodně mezi 1 a n, pak pravděpodobnost, že p je prvočíslo $\sim \frac{1}{\ln n}$

Logická otázka

"Kolik náhodných přirozených čísel je třeba otestovat, než najdeme prvočíslo?"

Definice: $\pi(n) =$ počet prvočísel $\leq n$

Prime Number Theorem: $\pi(n) \sim \frac{n}{\ln n}$

Důsledek: vybereme-li p náhodně mezi 1 a n, pak pravděpodobnost, že p je prvočíslo $\sim \frac{1}{\ln n}$

Příklad

Náhodné číslo o 512 bitech je prvočíslo s pravděpodobností $\sim \frac{1}{\ln 2^{512}} \doteq \frac{1}{355}$.

Stačí uvažovat jen lichá čísla $\Rightarrow$ *pravděpodobnost* $\sim \frac{2}{355}$.

Logická otázka

"Kolik náhodných přirozených čísel je třeba otestovat, než najdeme prvočíslo?"
Definice: $\pi(n) =$ počet prvočísel $\leq n$
Prime Number Theorem: $\pi(n) \sim \frac{n}{\ln n}$
Důsledek: vybereme-li p náhodně mezi 1 a n, pak pravděpodobnost, že p je prvočíslo $\sim \frac{1}{\ln n}$

Příklad

Náhodné číslo o 512 *bitech je prvočíslo s pravděpodobností* $\sim \frac{1}{\ln 2^{512}} \doteq \frac{1}{355}$.
Stačí uvažovat jen lichá čísla $\Rightarrow$ *pravděpodobnost* $\sim \frac{2}{355}$.

Deterministický test

1. **Primality Testing**
2. Factorization

Agrawal - Kalyan - Saxena

- 2002 - *PRIMES is in P*
- Manindra Agrawal, Neeraj Kayal, Nitin Saxena from the Indian Institute of Technology Kanpur
- složitost polynomiální $\mathcal{O}^{\sim}(\log^6 n)$


Agrawal


Kayal


Saxena

Hlavní myšlenka

Věta

Nechť $n \geq 2$ *a* $a \in \{1, 2, \ldots, n-1\}$ *nesoudělné s* n. *Pak* $n \in \mathbb{P} \Leftrightarrow$ $(x-a)^n \equiv x^n - a \bmod n$.

- výpočet n koeficientů $\Rightarrow$ náročné

Hlavní myšlenka

Věta

Nechť $n \geq 2$ *a* $a \in \{1, 2, \ldots, n-1\}$ *nesoudělné s* n. *Pak* $n \in \mathbb{P} \Leftrightarrow$ $(x-a)^n \equiv x^n - a \bmod n$.

- výpočet n koeficientů $\Rightarrow$ náročné
- **zjednodušení**: volba vhodného r tak, že (ne)splnění identity

$$(x-a)^n \equiv x^n - a \bmod (x^r - 1), n$$

pro málo hodnot a rozhodne o tom, zda $n \in \mathbb{P}$

Hlavní myšlenka

Věta

Nechť $n \geq 2$ *a* $a \in \{1, 2, \ldots, n-1\}$ *nesoudělné s* n. *Pak* $n \in \mathbb{P} \Leftrightarrow$ $(x-a)^n \equiv x^n - a \bmod n$.

- výpočet n koeficientů $\Rightarrow$ náročné
- **zjednodušení**: volba vhodného r tak, že (ne)splnění identity

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

pro málo hodnot a rozhodne o tom, zda $n \in \mathbb{P}$

Volba r

- chceme, aby multiplikativní řád $n \bmod r$ byl $> \log^2 n$
- najdeme minimální takové r

Volba r

- chceme, aby multiplikativní řád $n \bmod r$ byl $> \log^2 n$
- najdeme minimální takové r
- $B := \lceil \log^5 n \rceil$ a $A := n^{\lfloor \log B \rfloor} \prod_{i=1}^{\lfloor \log^2 n \rfloor} (n^i - 1)$

Volba r

- chceme, aby multiplikativní řád $n \bmod r$ byl $> \log^2 n$
- najdeme minimální takové r
- $B := \lceil \log^5 n \rceil$ a $A := n^{\lfloor \log B \rfloor} \prod_{i=1}^{\lfloor \log^2 n \rfloor} (n^i - 1)$
- $r :=$ nejmenší přirozené číslo, které nedělí A
- snadno se ověří, že
  1. $NSD(r, n) = 1$
  2. $r \leq B$
  3. $o_r(n) > \log^2 n$

Volba r

- chceme, aby multiplikativní řád $n \bmod r$ byl $> \log^2 n$
- najdeme minimální takové r
- $B := \lceil \log^5 n \rceil$ a $A := n^{\lfloor \log B \rfloor} \prod_{i=1}^{\lfloor \log^2 n \rfloor} (n^i - 1)$
- $r :=$ nejmenší přirozené číslo, které nedělí A
- snadno se ověří, že
  1. $NSD(r, n) = 1$
  2. $r \leq B$
  3. $o_r(n) > \log^2 n$

Algoritmus AKS

1. if $n = a^b$ pro $a \in \mathbb{N}$ a $b > 1 \Rightarrow n$ složené
2. najdi nejmenší r tak, že $o_r(n) > \log^2 n$

Algoritmus AKS

1. if $n = a^b$ pro $a \in \mathbb{N}$ a $b > 1 \Rightarrow n$ složené
2. najdi nejmenší r tak, že $o_r(n) > \log^2 n$
3. if $1 < NSD(a, n) < n$ pro nějaké $a \leq r \Rightarrow n$ složené

Algoritmus AKS

1. if $n = a^b$ pro $a \in \mathbb{N}$ a $b > 1 \Rightarrow n$ složené
2. najdi nejmenší r tak, že $o_r(n) > \log^2 n$
3. if $1 < NSD(a, n) < n$ pro nějaké $a \leq r \Rightarrow n$ složené
4. if $n \leq r \Rightarrow n$ prvočíslo

Algoritmus AKS

1. if $n = a^b$ pro $a \in \mathbb{N}$ a $b > 1 \Rightarrow n$ složené
2. najdi nejmenší r tak, že $o_r(n) > \log^2 n$
3. if $1 < NSD(a, n) < n$ pro nějaké $a \leq r \Rightarrow n$ složené
4. if $n \leq r \Rightarrow n$ prvočíslo
5. for $a = 1$ to $\lfloor \sqrt{\varphi(r)} \log n \rfloor$
   if $(x - a)^n \not\equiv x^n - a \bmod n(x^r - 1), n \Rightarrow n$ složené

Algoritmus AKS

1. if $n = a^b$ pro $a \in \mathbb{N}$ a $b > 1 \Rightarrow n$ složené
2. najdi nejmenší r tak, že $o_r(n) > \log^2 n$
3. if $1 < NSD(a, n) < n$ pro nějaké $a \leq r \Rightarrow n$ složené
4. if $n \leq r \Rightarrow n$ prvočíslo
5. for $a = 1$ to $\lfloor \sqrt{\varphi(r)} \log n \rfloor$
   if $(x - a)^n \not\equiv x^n - a \bmod n(x^r - 1), n \Rightarrow n$ složené
6. n prvočíslo

Algoritmus AKS

1. if $n = a^b$ pro $a \in \mathbb{N}$ a $b > 1 \Rightarrow n$ složené
2. najdi nejmenší r tak, že $o_r(n) > \log^2 n$
3. if $1 < NSD(a, n) < n$ pro nějaké $a \leq r \Rightarrow n$ složené
4. if $n \leq r \Rightarrow n$ prvočíslo
5. for $a = 1$ to $\lfloor \sqrt{\varphi(r)} \log n \rfloor$
   if $(x - a)^n \not\equiv x^n - a \bmod n(x^r - 1), n \Rightarrow n$ složené
6. n prvočíslo

Náznak důkazu - sporem

- předpokládejme, že n je složené a přesto output prvočíslo $\Rightarrow$ konec krokem 6
- uvažujme p/n a $p > r$ a $\ell := \lfloor \sqrt{\varphi(r)} \log n \rfloor$

Náznak důkazu - sporem

- předpokládejme, že n je složené a přesto output prvočíslo $\Rightarrow$ konec krokem 6
- uvažujme p/n a $p > r$ a $\ell := \lfloor \sqrt{\varphi(r)} \log n \rfloor$
- množina G prvků $k \bmod r$, pro které $(x-a)^k \equiv x^k - a \bmod (x^r - 1), n$ pro všechna $a < \ell$ tvoří grupu o velikosti aspoň $\log^2 n$

Náznak důkazu - sporem

- předpokládejme, že n je složené a přesto output prvočíslo $\Rightarrow$ konec krokem 6
- uvažujme p/n a $p > r$ a $\ell := \lfloor \sqrt{\varphi(r)} \log n \rfloor$
- množina G prvků $k \bmod r$, pro které $(x-a)^k \equiv x^k - a \bmod (x^r - 1), n$ pro všechna $a < \ell$ tvoří grupu o velikosti aspoň $\log^2 n$
- množina produktů $(x-a)^k \bmod h(x), p$ pro $a < \ell$, kde $h(x)$ je vhodně zvolený ireducibilní faktor $x^r - 1$, je také grupa $\mathcal{G}$

Náznak důkazu - sporem

- předpokládejme, že n je složené a přesto output prvočíslo $\Rightarrow$ konec krokem 6
- uvažujme p/n a $p > r$ a $\ell := \lfloor \sqrt{\varphi(r)} \log n \rfloor$
- množina G prvků $k \bmod r$, pro které $(x-a)^k \equiv x^k - a \bmod (x^r - 1), n$ pro všechna $a < \ell$ tvoří grupu o velikosti aspoň $\log^2 n$
- množina produktů $(x-a)^k \bmod h(x), p$ pro $a < \ell$, kde $h(x)$ je vhodně zvolený ireducibilní faktor $x^r - 1$, je také grupa $\mathcal{G}$
- lze určit horní a dolní odhady na velikost grupy $\mathcal{G}$

$$\binom{\#G + \ell}{\#G - 1} \leq \#\mathcal{G} \leq n^{\sqrt{\#G}},$$

které se ale vylučují, pokud n není p^b pro nějaké $b > 1 \Rightarrow$ SPOR

Náznak důkazu - sporem

- předpokládejme, že n je složené a přesto output prvočíslo $\Rightarrow$ konec krokem 6
- uvažujme p/n a $p > r$ a $\ell := \lfloor \sqrt{\varphi(r)} \log n \rfloor$
- množina G prvků $k \bmod r$, pro které $(x-a)^k \equiv x^k - a \bmod (x^r - 1), n$ pro všechna $a < \ell$ tvoří grupu o velikosti aspoň $\log^2 n$
- množina produktů $(x-a)^k \bmod h(x), p$ pro $a < \ell$, kde $h(x)$ je vhodně zvolený ireducibilní faktor $x^r - 1$, je také grupa $\mathcal{G}$
- lze určit horní a dolní odhady na velikost grupy $\mathcal{G}$

$$\left(\begin{array}{c} \#G + \ell \\ \#G - 1 \end{array} \right) \leq \#\mathcal{G} \leq n^{\sqrt{\#G}},$$

které se ale vylučují, pokud n není p^b pro nějaké $b > 1 \Rightarrow$ SPOR

Složitost algoritmu

- nejdelší částí je ověření, zda $(x-a)^n \equiv x^n - a \bmod (x^r - 1), n$ pro všechna $a \leq l$
- jednoduchá analýza $\Rightarrow \mathcal{O}^{\sim}(\log^{21/2} n)$

Složitost algoritmu

- nejdelší částí je ověření, zda $(x-a)^n \equiv x^n - a \bmod (x^r - 1), n$ pro všechna $a \leq l$
- jednoduchá analýza $\Rightarrow \mathcal{O}^{\sim}(\log^{21/2} n)$
- jemnější analýza $\Rightarrow \mathcal{O}^{\sim}(\log^{15/2} n)$

Složitost algoritmu

- nejdelší částí je ověření, zda $(x-a)^n \equiv x^n - a \bmod (x^r-1), n$ pro všechna $a \leq l$
- jednoduchá analýza $\Rightarrow \mathcal{O}^{\sim}(\log^{21/2} n)$
- jemnější analýza $\Rightarrow \mathcal{O}^{\sim}(\log^{15/2} n)$
- se změnou v algoritmu $\Rightarrow \mathcal{O}^{\sim}(\log^{6} n)$

Složitost algoritmu

- nejdelší částí je ověření, zda $(x-a)^n \equiv x^n - a \bmod (x^r-1), n$ pro všechna $a \leq l$
- jednoduchá analýza $\Rightarrow \mathcal{O}^{\sim}(\log^{21/2} n)$
- jemnější analýza $\Rightarrow \mathcal{O}^{\sim}(\log^{15/2} n)$
- se změnou v algoritmu $\Rightarrow \mathcal{O}^{\sim}(\log^{6} n)$
- stále mnohem pomalejší než Rabin-Miller

Složitost algoritmu

- nejdelší částí je ověření, zda $(x-a)^n \equiv x^n - a \bmod (x^r-1), n$ pro všechna $a \leq l$
- jednoduchá analýza $\Rightarrow \mathcal{O}^{\sim}(\log^{21/2} n)$
- jemnější analýza $\Rightarrow \mathcal{O}^{\sim}(\log^{15/2} n)$
- se změnou v algoritmu $\Rightarrow \mathcal{O}^{\sim}(\log^{6} n)$
- stále mnohem pomalejší než Rabin-Miller

Faktorizace

1. Primality Testing
2. **Factorization**

Pollard ρ metoda

- nechť n složené s prvočíselným faktorem p
- najdeme-li $a \equiv b \bmod p$ a $a \not\equiv b \bmod n$, pak $NSD(a-b, n)$ je netriviální faktor n

Pollard ρ metoda

- nechť n složené s prvočíselným faktorem p
- najdeme-li $a \equiv b \bmod p$ a $a \not\equiv b \bmod n$, pak $NSD(a - b, n)$ je netriviální faktor n

"Jak takový pár najít?"

Pollard ρ metoda

- nechť n složené s prvočíselným faktorem p
- najdeme-li $a \equiv b \bmod p$ a $a \not\equiv b \bmod n$, pak $NSD(a-b, n)$ je netriviální faktor n

"Jak takový pár najít?"

Odpověď: Iterováním náhodných funkcí

- nechť $f : \hat{n} \longrightarrow \hat{n}$ náhodná a x_0 náhodná počáteční hodnota
- definujme $x_{k+1} := f(x_k)$

Odpověď: Iterováním náhodných funkcí

- nechť $f : \hat{n} \longrightarrow \hat{n}$ náhodná a x_0 náhodná počáteční hodnota
- definujme $x_{k+1} := f(x_k)$
- maximání perioda poslupnosti (x_k) je délky n

Odpověď: Iterováním náhodných funkcí

- nechť $f : \hat{n} \longrightarrow \hat{n}$ náhodná a x_0 náhodná počáteční hodnota
- definujme $x_{k+1} := f(x_k)$
- maximání perioda poslupnosti (x_k) je délky n

"Jaká je obvyklá perioda?"

Odpověď: Iterováním náhodných funkcí

- nechť $f : \hat{n} \longrightarrow \hat{n}$ náhodná a x_0 náhodná počáteční hodnota
- definujme $x_{k+1} := f(x_k)$
- maximání perioda poslupnosti (x_k) je délky n

"Jaká je obvyklá perioda?"

Odpověď: Průměrná perioda je řádu $\sqrt{n}$

Birthday paradox (23 přátel na tahu má 50% šanci, že 2 z nich mají narozeniny ve stejný den)

Věta

Nechť $\lambda > 0$ a $\ell = 1 + [\sqrt{2\lambda n}]$. Pravděpodobnost, že $x_0, x_1, \ldots, x_\ell$ vzájemně různé $< e^{-\lambda}$.

Odpověď: Průměrná perioda je řádu $\sqrt{n}$

Birthday paradox (23 přátel na tahu má 50% šanci, že 2 z nich mají narozeniny ve stejný den)

Věta

Nechť $\lambda > 0$ *a* $\ell = 1 + [\sqrt{2\lambda n}]$. *Pravděpodobnost, že* $x_0, x_1, \ldots, x_\ell$ *vzájemně různé* $< e^{-\lambda}$.

Důležité: Lze očekávat, že posloupnost $(x_k \mod p)$ bude mít značně kratší periodu!

Odpověď: Průměrná perioda je řádu $\sqrt{n}$

Birthday paradox (23 přátel na tahu má 50% šanci, že 2 z nich mají narozeniny ve stejný den)

Věta

Nechť $\lambda > 0$ *a* $\ell = 1 + [\sqrt{2\lambda n}]$. *Pravděpodobnost, že* $x_0, x_1, \ldots, x_\ell$ *vzájemně různé* $< e^{-\lambda}$.

Důležité: Lze očekávat, že posloupnost $(x_k \mod p)$ bude mít značně kratší periodu!

Floyd's cycle trick

Odpověď: Průměrná perioda je řádu $\sqrt{n}$

Birthday paradox (23 přátel na tahu má 50% šanci, že 2 z nich mají narozeniny ve stejný den)

Věta

Nechť $\lambda > 0$ *a* $\ell = 1 + [\sqrt{2\lambda n}]$. *Pravděpodobnost, že* $x_0, x_1, \ldots, x_\ell$ *vzájemně různé* $< e^{-\lambda}$.

Důležité: Lze očekávat, že posloupnost $(x_k \mod p)$ bude mít značně kratší periodu!

Floyd's cycle trick

Hledání cyklů

- spočti $NSD(x_k - x_i, n)$ pro všechna $i < j \Rightarrow$ náročné!
- zlepšovák: porovnej x_k pouze s x_{2^i}, kde $2^i < k$ s i maximálním

Hledání cyklů

- spočti $NSD(x_k - x_i, n)$ pro všechna $i < j \Rightarrow$ náročné!
- zlepšovák: porovnej x_k pouze s x_{2^i}, kde $2^i < k$ s i maximálním
  1. x_2 s x_1,

Hledání cyklů

- spočti $NSD(x_k - x_i, n)$ pro všechna $i < j \Rightarrow$ náročné!
- zlepšovák: porovnej x_k pouze s x_{2^i}, kde $2^i < k$ s i maximálním
  1. x_2 s x_1,
  2. x_3 a x_4 s x_2,

Hledání cyklů

- spočti $NSD(x_k - x_i, n)$ pro všechna $i < j \Rightarrow$ náročné!
- zlepšovák: porovnej x_k pouze s x_{2^i}, kde $2^i < k$ s i maximálním
  1. x_2 s x_1,
  2. x_3 a x_4 s x_2,
  3. x_5 až x_8 s x_4 atd.

Hledání cyklů

- spočti $NSD(x_k - x_i, n)$ pro všechna $i < j \Rightarrow$ náročné!
- zlepšovák: porovnej x_k pouze s x_{2^i}, kde $2^i < k$ s i maximálním
  1. x_2 s x_1,
  2. x_3 a x_4 s x_2,
  3. x_5 až x_8 s x_4 atd.

  $\Rightarrow$ zpomalení nalezení periody mod p, ale méně výpočtů

Hledání cyklů

- spočti $NSD(x_k - x_i, n)$ pro všechna $i < j \Rightarrow$ náročné!
- zlepšovák: porovnej x_k pouze s x_{2^i}, kde $2^i < k$ s i maximálním
  1. x_2 s x_1,
  2. x_3 a x_4 s x_2,
  3. x_5 až x_8 s x_4 atd.

  $\Rightarrow$ zpomalení nalezení periody mod p, ale méně výpočtů

Algoritmus Pollard ρ metody

- faktorizujeme n, $f(x) := x^2 + 1$, náhodné $x_0 \in \{1, \ldots, n-1\}$
- spočteme $x_{k+1} = f(x_k) \bmod n$

Algoritmus Pollard ρ metody

- faktorizujeme n, $f(x) := x^2 + 1$, náhodné $x_0 \in \{1, \ldots, n-1\}$
- spočteme $x_{k+1} = f(x_k) \bmod n$
- pro $2^i = \max\{2^h | 2^h < k\}$ spočteme $NSD(x_k - x_{2^i}, n)$

Algoritmus Pollard ρ metody

- faktorizujeme n, $f(x) := x^2 + 1$, náhodné $x_0 \in \{1, \ldots, n-1\}$
- spočteme $x_{k+1} = f(x_k) \bmod n$
- pro $2^i = \max\{2^h | 2^h < k\}$ spočteme $NSD(x_k - x_{2^i}, n)$
- pokračujeme, dokud $NSD(x_k - x_{2^i}, n)$ není netriviální faktor n

Algoritmus Pollard ρ metody

- faktorizujeme n, $f(x) := x^2 + 1$, náhodné $x_0 \in \{1, \ldots, n-1\}$
- spočteme $x_{k+1} = f(x_k) \bmod n$
- pro $2^i = \max\{2^h | 2^h < k\}$ spočteme $NSD(x_k - x_{2^i}, n)$
- pokračujeme, dokud $NSD(x_k - x_{2^i}, n)$ není netriviální faktor n
- případně zvolíme jinou počáteční hodnotu x_0

Algoritmus Pollard ρ metody

- faktorizujeme n, $f(x) := x^2 + 1$, náhodné $x_0 \in \{1, \ldots, n-1\}$
- spočteme $x_{k+1} = f(x_k) \bmod n$
- pro $2^i = \max\{2^h | 2^h < k\}$ spočteme $NSD(x_k - x_{2^i}, n)$
- pokračujeme, dokud $NSD(x_k - x_{2^i}, n)$ není netriviální faktor n
- případně zvolíme jinou počáteční hodnotu x_0

Věta

Nechť n složené a p prvočíselný faktor $< \sqrt{n}$. Pollard ρ metoda najde netriviálního dělitele s velkou pravděpodobností v čase $\mathcal{O}(n^{1/4} \log^3 n)$.

Algoritmus Pollard ρ metody

- faktorizujeme n, $f(x) := x^2 + 1$, náhodné $x_0 \in \{1, \ldots, n-1\}$
- spočteme $x_{k+1} = f(x_k) \bmod n$
- pro $2^i = \max\{2^h | 2^h < k\}$ spočteme $NSD(x_k - x_{2^i}, n)$
- pokračujeme, dokud $NSD(x_k - x_{2^i}, n)$ není netriviální faktor n
- případně zvolíme jinou počáteční hodnotu x_0

Věta

Nechť n složené a p prvočíselný faktor $< \sqrt{n}$. Pollard ρ metoda najde netriviálního dělitele s velkou pravděpodobností v čase $\mathcal{O}(n^{1/4} \log^3 n)$.

Přesněji: existuje C tak, že v $C\sqrt{\lambda} n^{1/4} \log^3 n$ krocích metoda faktor najde s pravděpodobností $\geq 1 - e^{-\lambda}$

Algoritmus Pollard ρ metody

- faktorizujeme n, $f(x) := x^2 + 1$, náhodné $x_0 \in \{1, \ldots, n-1\}$
- spočteme $x_{k+1} = f(x_k) \bmod n$
- pro $2^i = \max\{2^h | 2^h < k\}$ spočteme $NSD(x_k - x_{2^i}, n)$
- pokračujeme, dokud $NSD(x_k - x_{2^i}, n)$ není netriviální faktor n
- případně zvolíme jinou počáteční hodnotu x_0

Věta

Nechť n složené a p prvočíselný faktor $< \sqrt{n}$. Pollard ρ metoda najde netriviálního dělitele s velkou pravděpodobností v čase $\mathcal{O}(n^{1/4} \log^3 n)$.

Přesněji: existuje C tak, že v $C\sqrt{\lambda} n^{1/4} \log^3 n$ krocích metoda faktor najde s pravděpodobností $\geq 1 - e^{-\lambda}$

Příklad: Pollard ρ metoda

Příklad

$n = 4171$, $x_0 = 2$, $f(x) = x^2 + 1$

- $x_1 = 5$, $x_2 = 26$, $NSD(26 - 5, 4171) = 1$
- $x_3 = 677$, $NSD(677 - 26, 4171) = 1$
- $x_4 = 3691$, $NSD(3691 - 26, 4171) = 1$
- $x_5 = 996$, $NSD(996 - 3691, 4171) = 1$
- $x_6 = 3490$, $NSD(3490 - 3691, 4171) = 1$
- $x_7 = 781$, $NSD(781 - 3691, 4171) = 97$
- *Poznámka:* $x_8 = 996 = x_5$