Matematické základy kryptografických algoritmů

Eliška Ochodková

Text byl vytvořen v rámci realizace projektu *Matematika pro inženýry 21. století* (reg. č. CZ.1.07/2.2.00/07.0332), na kterém se společně podílela Vysoká škola báňská – Technická univerzita Ostrava a Západočeská univerzita v Plzni

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Eliška Ochodková
Matematické základy kryptografických algoritmů

ISBN

Předmluva

Vážený čtenáři,

text, který právě čtete, vznikl v rámci řešení projektu "Matematika pro inženýry 21. století – inovace výuky matematiky na technických školách v nových podmínkách rychle se vyvíjející informační a technické společnosti". Projekt je řešen na Vysoké škole báňské - Technické univerzitě v Ostravě a Západočeské univerzitě v Plzni v období 2009 – 2012.

Hlavní motivací projektu je potřeba reagovat na změny významu jednotlivých partií matematiky při řešení praktických problémů, způsobenou zejména velkým pokrokem v matematickém modelování, dramatickým zlepšováním software a rychlým zvyšováním výpočetních kapacit moderních počítačů. Inženýři nyní běžně využívají stále se vyvíjející komplikované softwarové produkty založené na matematických pojmech, se kterými se v kurzech matematiky buďto nesetkají vůbec nebo v nevhodné formě. Na druhé straně prezentace některých pojmů v základních kurzech neodráží z nejrůznějších důvodů potřeby odborných kateder. Bohužel tento stav ztěžuje studentům aktivní používání získaných vědomostí v odborných předmětech i orientaci v rychle se vyvíjejících metodách inženýrské praxe.

Cílem projektu je inovace matematických a některých odborných kurzů na technických vysokých školách s cílem získat zájem studentů, zvýšit efektivnost výuky, zpřístupnit prakticky aplikovatelné výsledky moderní matematiky a vytvořit předpoklady pro efektivní výuku inženýrských předmětů. Zkvalitnění výuky matematiky budoucích inženýrů chceme dosáhnout po stránce formální využitím nových informačních technologií přípravy elektronických studijních materiálů a po stránce věcné pečlivým výběrem vyučované látky s důsledným využíváním zavedených pojmů v celém kurzu matematiky s promyšlenou integrací moderního matematického aparátu do vybraných inženýrských předmětů. Metodiku výuky matematiky a její atraktivnost pro studenty chceme zlepšit důrazem na motivaci a důsledným používáním postupu "od problému k řešení".

V rámci projektu vytváříme 40 nových výukových materiálů z oblastí matematické analýzy, lineární algebry, numerických metod, metod optimalizace, diskrétní matematiky, teorie grafů, statistiky a několika odborných kurzů. Všechny hotové výukové materiály budou volně k dispozici na webových stránkách projektu http://mi21.vsb.cz

Autoři předem děkují za všechny případné nápady a návrhy k vylepšení textu i za upozornění na chyby.

Jeden z obtížných momentů při tvorbě skript bylo rozhodnutí o šíři probírané problematiky a o tom, do jakých detailů při jejím výkladu jít. Kryptografie a kryptografické algoritmy jsou téma velmi obsáhlé. Tento text tak neobsahuje popis jednotlivých algoritmů, nezabývá se jejich aplikací (například v bezpečnostních protokolech) a nepostihuje vyčerpávajícím způsobem veškerý matematický základ, použitý v některém z algoritmů. Výběr témat byl nakonec učiněn na základě zkušeností z výuky předmětu Kryptografie a počítačová bezpečnost na FEI VŠB - TU Ostrava. Autorka předpokládá, že v budoucnu na tento text naváže další, nebo bude tento rozšířen o další témata.

Na závěr zbývá uvést, že veškeré informace byly čerpány z publikací [2, 5, 7, 8, 9, 10]. Na tyto prameny nejsou v textu odkazy uváděny, protože by jich bylo jednak příliš mnoho, a přece jen, většina textu se týká obecně známých poznatků. Na méně známá fakta jsou odkazy v textu uvedeny.

Obsah

Předmluva **iii**

1 Matematický základ **1**
1.1 Základní pojmy a značení . . . 1
1.2 Úvod do teorie čísel . . . 5
Příklady k procvičení . . . 7
1.3 Modulární aritmetika . . . 8
Příklady k procvičení . . . 11

2 Složitost **12**
2.1 Úvod do teorie složitosti . . . 12
2.2 Složitostní míry . . . 15
Příklady k procvičení . . . 17

3 Algebraické struktury **20**
3.1 Grupy . . . 20
Příklady k procvičení . . . 23
3.2 Tělesa . . . 25
3.3 Konečná tělesa . . . 27
3.3.1 Konečná tělesa $GF(p)$ . . . 27
3.3.2 Konečná tělesa $GF(2^m)$ . . . 28
Příklady k procvičení . . . 32

4 Algoritmy **35**
4.1 Euklidův algoritmus . . . 35
4.2 Rozšířený Euklidův algoritmus . . . 37
Příklady k procvičení . . . 38
4.3 Rychlé modulární umocňování . . . 39
Příklady k procvičení . . . 40

5 Testování prvočíselnosti **41**
5.1 Nejjednodušší algoritmy . . . 42
5.2 Pravděpodobnostní testy . . . 43
5.2.1 Fermatův test . . . 44

5.2.2 Miller-Rabinův test 46
5.3 Testy dokazující prvočíselnost 50
Příklady k procvičení 51

6 Důležité věty a problémy **52**
6.1 Malá Fermatova věta 52
6.2 Eulerova věta 54
Příklady k procvičení 56
6.3 Čínská věta o zbytcích 57
Příklady k procvičení 59
6.4 Problém diskrétního logaritmu 60
Příklady k procvičení 63

7 Kryptografie na bázi eliptických křivek **66**
7.1 Pojem eliptické křivky 67
7.2 Operace sčítání bodů 70
7.3 Eliptické křivky nad $\mathbb{Z}_p$ 73
7.4 Eliptické křivky nad $GF(2^m)$ 76
7.5 Diskrétní logaritmus nad eliptickou křivkou 77
Příklady k procvičení 78

8 Výsledky cvičení **79**

Literatura **88**

Rejstřík **89**

Kapitola 1

Matematický základ

Mnoho kryptografických algoritmů využívá diskrétní množinu „čísel“ a s čísly manipuluje pomocí základních aritmetických operací sčítání a násobení. Algoritmy zpravidla pracují jednak s nezápornými celými čísly $0, 1, \ldots$ jednak s polynomy (např. $x^2 + x + 1, \ldots$). Obě operace sčítání i násobení jsou však prováděny modulo kladné celé číslo resp. modulo polynom. V této kapitole seznámíme čtenáře s problematikou dělitelnosti celých čísel, s relací kongruence a jejími vlastnostmi. Na začátku kapitoly shrneme pojmy, jejichž znalost se u čtenáře předpokládá.

1.1 Základní pojmy a značení

Úvodní část tohoto učebního textu je věnována shrnutí některých základních matematickým pojmů, jejichž znalost se u čtenáře předpokládá.

Kolekce objektů, které lze navzájem odlišit, nazýváme *množinou* a tyto objekty nazýváme *prvky* množiny. To, že objekt o je prvkem množiny M zapisujeme $o \in M$; skutečnost, že objekt o není prvkem množiny M, zapisujeme $o \notin M$. Vzájemný vztah dvou a více množin budeme značit obvyklým způsobem:

- $A \subseteq B$ znamená, že množina A je *podmnožinou* množiny B.
- $A \subset B$ znamená, že A je *vlastní podmnožinou* množiny B, tj. $A \subseteq B$ a $A \neq B$.
- *Průnik* množin A a B je množina $A \cap B = \{x \mid x \in A \text{ a } x \in B\}$.
- *Sjednocení množin* A a B je množina $A \cup B = \{x \mid x \in A \text{ nebo } x \in B\}$.
- *Rozdíl množin* A a B je množina $A \setminus B = \{x \mid x \in A \text{ a } x \notin B\}$.
- *Kartézský součin* množin A a B je množina $A \times B = \{(a, b) \mid a \in A \text{ a } b \in B\}$, kde (a, b) je uspořádaná dvojice prvků. Např. $\{a_1, a_2\} \times \{b_1, b_2, b_3\} = \{(a_1, b_1), (a_1, b_2), (a_1, b_3), (a_2, b_1), (a_2, b_2), (a_2, b_3)\}$.

- *Binární relace* R na množině A je libovolná podmnožina kartézského součinu $A \times A$, píšeme $R \subseteq A \times A$. Relace $R \subseteq A \times A$ je:
  - *reflexivní*, jestliže pro všechna $a \in A$ platí $(a, a) \in R$,
  - *symetrická*, jestliže pro všechna $a, b \in A$ platí, že pokud $(a, b) \in R$, pak $(b, a) \in R$,
  - *tranzitivní*, jestliže pro všechna $a, b, c \in A$ platí, že pokud $(a, b) \in R$ a $(b, c) \in R$, pak $(a, c) \in R$.

V celém textu budeme pracovat především:

- s množinou celých čísel $\mathbb{N} = \{1, 2, \ldots\}$,
- s množinou celých čísel $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$ resp. s podmnožinou $\mathbb{Z}$.

Dalším potřebným pojmem, který si připomeneme, je pojem funkce. *Funkce* (nebo zobrazení) f z množiny A do množiny B (píšeme $f : A \longrightarrow B$) přiřazuje každému $a \in A$ právě jeden prvek $b \in B$. Prvek b se nazývá obraz prvku a, prvek a se nazývá vzor prvku b, tuto skutečnost zapisujeme $f(a) = b$. Množinu A nazýváme definičním oborem funkce f a množinu B nazýváme oborem hodnot funkce f.

- A funkce $f : A \longrightarrow B$ je *prostá* neboli injektivní, jestliže každý prvek z B je obrazem nejvýše jednoho prvku z A. Proto, jestliže $f(a_1) = f(a_2)$, pak $a_1 = a_2$.
- Funkce $f : A \longrightarrow B$ je *na* neboli surjektivní, jestliže každé $b \in B$ je obrazem alespoň jednoho $a \in A$.
- Funkce $f : A \longrightarrow B$ je *vzájemně jednoznačná* (bijektivní), jestliže je současně injektivní i surjektivní. Je-li funkce f bijekcí mezi konečnými množinami A a B, potom $|A| = |B|$ (symbolem $|A|$ označujemem mohutnost (počet prvků) množiny).
- Jestliže je funkce f je bijekcí, pak je funkce *inverzní* f^{-1} k funkci f definována takto: $f^{-1}(b) = a$ právě když $f(a) = b$.

Jedním ze základních principů používaných v kryptografických šifrovacích algoritmech je přeskupení znaků zprávy. Toto přeskupení není ničím jiným než permutací (trenspozicí) znaků. Připomeňme si proto, co je permutace.

Permutací n-prvkové množiny X rozumíme libovolnou bijekci (vzájemně jednoznačné zobrazení) $f : X \to X$. Tuto bijekci můžeme zadávat pomocí tabulky se dvěma řádky tak, že každý řádek bude obsahovat všechny prvky množiny X, přičemž prvek $f(x)$ bude umístěn pod prvkem x.

Příklad 1.1. Například je-li $X = \{a, b, c, d\}$ a permutace $f : X \to X$ je zadaná následovně $f(a) = c$, $f(b) = b$, $f(c) = d$, $f(d) = a$, potom

$$f = \begin{pmatrix} a & b & c & d \\ c & b & d & a \end{pmatrix}$$

▲

Počet všech permutací n prvkové množiny je roven číslu $n!$. Rychlost růstu počtu permutací ukazuje tabulka 1.1.

n!	n
0	1
1	1
2	2
6	3
24	4
120	5
720	6
5 040	7
40 320	8
362 880	9
3 628 800	10
39 916 800	11
479 001 600	12
6 227 020 800	13
87 178 291 200	14
1 307 674 368 000	15
20 922 789 888 000	16
355 687 428 096 000	17
6 402 373 705 728 000	18
121 645 100 408 832 000	19
2 432 902 008 176 640 000	20

Tab. 1.1 Růst počtu permutací

Uvažujme nyní bez újmy na obecnosti permutace množiny $X = \{1, \ldots, n\}$. Označíme S_n množinu všech permutací množiny $\{1, \ldots, n\}$. Libovolnou permutaci $f \in S_n$ můžeme ztotožnit s posloupností $\langle a_1, \ldots, a_n \rangle$, kde $f(i) = a_i$. Součinem dvou permutací $f, g \in S_n$ budeme rozumět permutaci $f \circ g$ definovanou jako superpozice zobrazení f a g. To znamená, že $(f \circ g)(i) = f(g(i))$.

Příklad 1.2. Například jsou-li $f, g \in S_7$ takové, že

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 7 & 1 & 3 & 6 & 2 & 4 & 5 \end{pmatrix}$$

$$g = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 3 & 4 & 5 & 7 & 2 & 6 \end{pmatrix}$$

potom

$$f \circ g = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 7 & 3 & 6 & 2 & 5 & 1 & 4 \end{pmatrix}$$

▲

Permutaci

$$id = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{pmatrix}$$

nazveme *identickou permutací*. Je zřejmé, že platí $id \circ f = f \circ id = f$. Snadno se ukáže, že ke každé permutaci $f \in S_n$ existuje permutace $f^{-1} \in S_n$ taková, že $f \circ f^{-1} = f^{-1} \circ f = id$. Tuto permutaci budeme nazývat *inverzní permutací* k permutaci f. Inverzní permutaci k permutaci f dostaneme tak, že vyměníme řádky v zápisu permutace f.

Příklad 1.3. Například pro

$$f = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 7 & 1 & 3 & 6 & 2 & 4 & 5 \end{pmatrix}$$

dostaneme

$$f^{-1} = \begin{pmatrix} 7 & 1 & 3 & 6 & 2 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ 2 & 5 & 3 & 6 & 7 & 4 & 1 \end{pmatrix}$$

▲

Dále je platí, že pro libovolné permutace $f, g, h \in S_n$ platí následující identity:

- $(f \circ g) \circ h = f \circ (g \circ h)$
- $id \circ f = f \circ id = f$
- $f^{-1} \circ f = f \circ f^{-1} = id$.

To znamená, že S_n je grupa vzhledem k operaci součinu permutací, viz kapitola 3.1.

1.2 Úvod do teorie čísel

Teorie čísel je ta část matematiky, jejímž základním objektem zkoumání jsou vlastnosti zejména celých čísel. V této kapitole je uveden pomocný matematický aparát související s kryptografií.

Definice 1.4. Nechť a a b jsou celá. Potom číslo a dělí číslo b právě tehdy, když existuje celé číslo c takové, že $b = a \cdot c$. Číslo a se nazývá *dělitel* čísla b, zapisujeme $a \mid b$.

Mezi další známé pojmy a fakta patří:

- Jako *triviální dělitele* $a \in \mathbb{Z}$ označujeme čísla 1 a číslo a samotné.
- Celé číslo c je *společným dělitelem* c.č. a a b, jestliže $c \mid a$ a $c \mid b$.
- *Prvočíslo* je číslo $\geqq 2$, které nemá jiné dělitele než triviální.
- Číslo, které není prvočíslem, nazýváme *číslem složeným*.
- Každé celé číslo $\geqq 2$ je buď prvočíslo, nebo se dá zapsat jako součin prvočísel.

Definice 1.5. Jestliže $a, b \in \mathbb{Z}$ a $b \geqq 1$, pak dělení čísla a číslem b dává čísla q (*podíl*) a r (*zbytek*) taková, že $a = qb + r$, kde $0 \leqq r < b$. Čísla r a q jsou jednoznačná. Zbytek po dělení budeme značit také $a \pmod b$.

Definice 1.6. Každé celé číslo $n \geqq 2$ se dá zapsat jednoznačně (až na pořadí) v *kanonickém tvaru* $n = {p_1}^{a_1} \cdot {p_2}^{a_2} \cdot \ldots \cdot {p_k}^{a_k}$, kde $p_1, p_2, \ldots, p_k$ jsou navzájem různá prvočísla a $a_1, a_2, \ldots, a_k$ jsou kladná celá čísla.

Příklad 1.7. Příklady kanonického rozkladu:
$3600 = 2^4 \cdot 3^2 \cdot 5^2$,
$4864 = 2^8 \cdot 19$,
$3458 = 2 \cdot 7 \cdot 13 \cdot 19$.
▲

V kryptografii je často nutné pracovat s velkými prvočísly (jako např. v algoritmu RSA), a proto musíme testovat, zda je náhodně vygenerované číslo prvočíslem. Úloha zjistit, zda je dané celé číslo prvočíslem nebo číslem složeným, není triviální. Pro malá čísla (která mají cca 20 cifer) můžeme použít např. algoritmus Trial division, pro velká čísla (řádově 100 a více cifer) je to však postup nepoužitelný. Výklad některých algoritmů pro testování prvočíselnosti je předmětem kapitoly 5.

Definice 1.8. Kladné celé číslo d je *největším společným dělitelem* (dále jen NSD, anglicky GCD - greatest common divisor) celých čísel a, b, když platí:

a) d je společný dělitel celých čísel a, b.

b) Jestliže nějaké celé číslo d_1 dělí obě čísla a, b, pak d_1 dělí také d.

Příklad 1.9. Společní dělitelé čísel 12 a 18 jsou $\{\pm 1, \pm 2, \pm 3, \pm 6\}$. $\mathrm{NSD}(12, 18) = 6$.
▲

Definice 1.10. Kladné celé číslo d je *nejmenším společným násobkem* (dále jen NSN, anglicky LCM - least common multiple) celých čísel a, b když platí:

a) $a \mid d$ a $b \mid d$.

b) Jestliže $a \mid d_1$ a $b \mid d_1$, pak $d \mid d_1$ pro nějaké celé číslo d_1.

Příklad 1.11. Jestliže a a b jsou kladná celá čísla, pak $\mathrm{NSN}(a, b) = a \cdot b / \mathrm{NSD}(a, b)$. Např. $\mathrm{NSN}(12, 18) = 12 \cdot 18 / \mathrm{NSD}(12, 18) = 12 \cdot 3 = 36$.
▲

Příklad 1.12. Je snadné nalézt NSD a NSN dvou c. čísel $\geqq 2$, pokud je vyjádříme v kanonickém tvaru:
$300 = 2^2 \cdot 3^1 \cdot 5^2$,
$18 = 2^1 \cdot 3^2$,
$\mathrm{NSD}(300, 18) = 2^1 \cdot 3^1 \cdot 5^0 = 6$
$\mathrm{NSN}(300, 18) = 2^2 \cdot 3^2 \cdot 5^2 = 900$.
▲

Definice 1.13. Dvě celá čísla jsou *nesoudělná* (relatively prime), když jejich NSD je 1.

Příklad 1.14. Čísla 8 a 15 jsou nesoudělná, protože 8 má dělitele $1, 2, 4, 8$ a 15 má dělitele $1, 3, 5, 15$. Tedy jediným společným dělitelem je 1, $\mathrm{NSD}(8, 15) = 1$.
▲

Nejznámějším algoritmem pro nalezení NSD je Euklidův algoritmus, jehož princip si ozřejmíme v kapitole 4.

Definice 1.15. *Eulerova funkce* (Euler's totient function) $\phi(n)$ udává počet těch celých čísel $\geqq 1$ nepřesahujících kladné celé číslo $n \geqq 1$, která jsou nesoudělná s n.

Vlastnosti Eulerovy funkce:

- Jestliže n je prvočíslo, pak platí, že $\phi(n) = n - 1$, např. $\phi(19) = 18$.
- Eulerova funkce je multiplikativni funkce, tj. $\phi(m \cdot n) = \phi(m) \cdot \phi(n)$, pokud m a n jsou nesoudělná. Např. $\phi(21) = \phi(7) \cdot \phi(3) = 6 \cdot 2 = 12$.
- $\phi(n) = \phi(p^k) = (p-1) \cdot p^{k-1}$, pokud n je k-tou mocninou prvočísla p. Např. $\phi(25) = \phi(5^2) = (4) \cdot 5^{2-1} = 20$.
- obecně - mějme kanonický rozklad $n = {p_1}^{a_1} \cdot {p_2}^{a_2} \cdot \ldots \cdot {p_k}^{a_k}$, pak $\phi(n) = (p_1 - -1)p_1^{a_1-1} \cdot (p_2-1)p_2^{a_2-1} \cdot \ldots \cdot (p_k-1)p_k^{a_k-1}$, zapisuje se často jako $\phi(n) = n \cdot (1 - - \frac{1}{p_1}) \cdot (1 - \frac{1}{p_2}) \cdot \ldots \cdot (1 - \frac{1}{p_k})$.
  Např. $60 = 2^3 \cdot 3 \cdot 5$, $\phi(60) = 60 \cdot (1 - 1/2) \cdot (1 - 1/3) \cdot (1 - 1/5) = 16$.

Příklady k procvičení

1. Jaké dělitele má číslo 24? Jaké dělitele má číslo 29?

2. Určete kanonický rozklad čísla 356 a čísla 411.

3. Jsou čísla 120 a 49 nesoudělná?

4. Určete všechna celá kladná čísla menší než 26 a s 26 nesoudělná.

5. Nalezněte NSD$(270, 36)$ a NSN$(270, 36)$ pomocí kanonického rozkladu.

6. Nalezněte NSD$(1575, 23100)$ a NSN$(1575, 23100)$ pomocí kanonického rozkladu.

7. Určete $\phi(35)$, $\phi(64)$, $\phi(123)$, $\phi(600)$ a $\phi(13)$.

8. Dokažte následující tvrzení o $\phi(n)$.
   - Jestliže n je liché, pak $\phi(2n) = \phi(n)$.
   - Jestliže n je sudé, pak $\phi(2n) = 2\phi(n)$.
   - Jestliže $n > 2$ pak $\phi(n)$ je sudé.
   - Jestliže $d \mid n$ pak $\phi(d) \mid \phi(n)$.

1.3 Modulární aritmetika

Tato podkapitola se věnuje *modulární aritmetice*, tedy aritmetice, která provádí operace modulo nějaké (kladné) celé číslo. Protože množina celých čísel není uzavřená vzhledem k dělení nenulovými celými čísly, zavedeme pro další výklad tzv. celočíselný podíl. Připomeňme, že jestliže x je reálné číslo, potom $\lfloor x \rfloor$ je největší celé číslo menší nebo rovno x (tzv. *dolní celá část* čísla x). Dále předpokládejme, že n je kladné celé číslo.

Definice 1.16. Jestliže a je celé číslo a n je kladné celé číslo, pak definujeme $a \pmod n$ jako *zbytek po dělení* čísla a číslem n (číslo n se nazývá *modul* (nebo také modulus)). Pro každé a můžeme psát $a = \lfloor a/n \rfloor \cdot n + a \pmod n$.

Definice 1.17. Celé číslo a je *kongruentní modulo* n s celým číslem b tehdy, když je rozdíl $a - b$ dělitelný číslem n. Píšeme $a \equiv b \pmod n$.

Příklad 1.18. $a = \lfloor a/n \rfloor \cdot n + a \pmod n$, pro $a = 11, n = 7$ máme $11 = \lfloor 11/7 \rfloor \cdot 7 + 1 \pmod 7$,
$13 \equiv -9 \pmod{11}$ protože $13 - (-9) = 2 \cdot 11$,
$13 \not\equiv -9 \pmod 7$ protože $13 - (-9) = 2 \cdot 11 = 22$, což není číslo dělitelné 7.
▲

Poznámka 1.19. (Vlastnosti kongruence) Pro všechna $a, b, a_1, b_1, c \in \mathbb{Z}$ platí:

- $a \equiv b \pmod n$ právě tehdy, když zbytek po dělení čísel a i b číslem n je stejný.
- Reflexivita: $a \equiv a \pmod n$ pro $\forall a \in \mathbb{Z}$,
- Symetrie: Jestliže $a \equiv b \pmod n \Longrightarrow b \equiv a \pmod n$,
- Tranzitivita: jestliže $a \equiv b \pmod n$ a $b \equiv c \pmod n$, potom $a \equiv c \pmod n$.
- Jestliže $a \equiv a_1 \pmod n$ a $b \equiv b_1 \pmod n$, potom $a + b \equiv a_1 + b_1 \pmod n$.

Z předchozího vidíme, že relace kongruence je *ekvivalencí*, protože je reflexivní, symetrická a tranzitivní. Třída ekvivalence c.č. a je množina všech celých čísel kongruentních s a modulo n. Vidíme, že pro dané n relace kongruence a modulo n dělí množinu $\mathbb{Z}$ na třídy ekvivalence, na tzv. zbytkové třídy modulo n.

Nyní, je-li $a = qn + r$, kde $0 \leqq r < n$, pak $a \equiv r \pmod n$. Tj. každé celé číslo. a je (jednoznačně) kongruentní *mod* n s kladným celým číslem z intervalu 0 až $n - 1$. Čísla a a r patří do téže třídy ekvivalence a r můžeme použít jako reprezentanta této třídy ekvivalence. Celá čísla modulo n jsou množinou (tříd ekvivalence) celých čísel $\{0, 1, \ldots, n-1\}$, ozn. $\mathbb{Z}_n$. Operace sčítání a násobení v $\mathbb{Z}_n$ jsou prováděny modulo n.

Definice 1.20. Nechť je dáno $n > 0$. Pro každé celé číslo a definujeme $[a]_n = \{x \mid x \equiv a \pmod{n}\}$ jako množinu celých čísel kongruentních a modulo n a nazveme ji *množinou zbytkových tříd modulo n*.

Příklad 1.21. Každé číslo ze $\mathbb{Z}_n$ reprezentuje zbytkovou třídu. Nechť $n = 7$, zbytkové třídy modulo 7 označme jako $[0]_7, [1]_7, \ldots, [7-1]_7$, pak $[2]_7 = \{\ldots, -12, -5, 2, 9, 16, \ldots\}$. Všechna čísla ve třetím sloupci jsou ekvivalentní (mají stejný zbytek):

$$
\begin{aligned}
-12 &= -2 \cdot 7 + 2 \\
-5 &= -1 \cdot 7 + 2 \\
2 &= 0 \cdot 7 + 2 \\
9 &= 1 \cdot 7 + 2 \\
16 &= 2 \cdot 7 + 2
\end{aligned}
$$

-14	-13	-12	-11	-10	-9	-8
-7	-6	-5	-4	-3	-2	-1
0	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20

▲

Poznámka 1.22. Pro operace $(+)$ a $(\cdot)$ modulo n platí:

- Výsledkem $a + b$ bude číslo $c \in \mathbb{Z}_n$ pro které platí $a + b \equiv c \pmod{n}$.
- Výsledkem $a \cdot b$ bude číslo $c \in \mathbb{Z}_n$ pro které platí $a \cdot b \equiv c \pmod{m}$.
- Např. nechť $n = 9, \mathbb{Z}_n = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$, pak $6 + 8 = 14 \equiv 5 \pmod{9}$, $6 \cdot 8 = 48 \equiv 3 \pmod{9}$.

Definice 1.23. Nechť $a \in \mathbb{Z}_n$. *Multiplikativní inverzní prvek* k prvku a modulo n je číslo $x \in \mathbb{Z}_n$ takové, že $a \cdot x \equiv 1 \pmod{n}$. Pokud takové x existuje, je jednoznačné. Inverzní prvek k a modulo n budeme označovat a^{-1}.

Definice 1.24. Nechť $a \in \mathbb{Z}_n$. Dělení čísla a číslem b modulo n je definováno jako součin $a \cdot b^{-1}$ modulo n a je definováno jen tehdy, je-li b *invertovatelné mod n*.

Definice 1.25. Nechť $a \in \mathbb{Z}_n$. *Aditivní opačný prvek* k prvku a modulo n je číslo $x \in \mathbb{Z}_n$ takové, že $a + x \equiv 0 \pmod{n}$. Pokud takové x existuje, je jednoznačné. Opačný prvek k a modulo n budeme označovat $-a$.

Pro určení toho, zda k danému $a \in \mathbb{Z}_n$ existuje inverzní prvek využijeme faktu, že číslo a invertovatelné právě tehdy když $\mathrm{NSD}(a, n) = 1$. Praktické aplikace pak pro nalezení multiplikativního inverzního prvku používají tzv. rozšířený Euklidův algoritmus (viz kap. 4).

Příklad 1.26. Nechť $n = 9, \mathbb{Z}_n = \{0, 1, 2, 3, 4, 5, 6, 7, 8\}$.
Invertovatelné jsou prvky $1, 2, 4, 5, 7, 8$, např. $5^{-1} = 2$, protože $5 \cdot 2 \equiv 1 \pmod{9}$.
▲

Poznámka 1.27. Některé další vlastnosti modulární aritmetiky:

- Platí: $[(a \pmod n) + (b \pmod n)] \pmod n = (a + b) \pmod n$
  Např.: $[(11 \pmod 8) + (15 \pmod 8)] \pmod 8 = 10 \pmod 8 = 2$,
  $(11 + 15) \pmod 8 = 26 \pmod 8 = 2$.

- Platí: $[(a \pmod n) \cdot (b \pmod n)] \pmod n = (a \cdot b) \pmod n$
  Např.: $[(11 \pmod 8) \cdot (15 \pmod 8)] \pmod 8 = 21 \pmod 8 = 5$,
  $(11 \cdot 15) \pmod 8 = 165 \pmod 8 = 5$.

- Dále jestliže $(a + b) \equiv (a + c) \pmod n$ pak $b \equiv c \pmod n$.
  Např. jestliže $5 + 23 \equiv (5 + 7) \pmod 8$ pak $23 \equiv 7 \pmod 8$.
  Přidáme-li opačný prvek vzhledem k operaci $(+)$ k oběma stranám dostaneme
  $((-a) + a + b) \equiv ((-a) + a + c) \pmod n$ pak $b \equiv c \pmod n$.

- Jestliže $(a \cdot b) \pmod n \equiv (a \cdot c) \pmod n$ pak $b \equiv c \pmod n$ platí jen tehdy když a je nesoudělné s n.
  Např.: platí $6 \cdot 3 = 18 \equiv 2 \pmod 8$ a $6 \cdot 7 = 42 \equiv 2 \pmod 8$
  ale $3 \not\equiv 7 \pmod 8$, protože $\mathrm{NSD}(6, 8) \neq 1$, tj. jsou to čísla soudělná.
  Pokud jsou a a n nesoudělná, přidáme-li inverzní prvek prvek vzhledem k operaci $(\cdot)$ k oběma stranám, dostaneme $((a^{-1}) \cdot a \cdot b) \pmod n \equiv ((a^{-1}) \cdot a \cdot c) \pmod n$ pak $b \equiv c \pmod n$.

- Na modulární umocňování se můžeme dívat jako na opakované násobení. Pro určení $11^7 \pmod{13}$, postupujeme takto:
  $11^2 = 121 \equiv 4 \pmod{13}$
  $11^4 = 11^2 \cdot 11^2 \equiv 4^2 \equiv 3 \pmod 1 3$
  $11^7 = 11^1 \cdot 11^2 \cdot 11^4 \equiv 11 \cdot 4 \cdot 3 = 132 \equiv 2 \pmod{13}$.
  Pro velká čísla se pro umocňování používá algoritmus Repeated square and multiply algorithm, viz kapitola 4.

Příklad 1.28. Kolik je $3^8 \pmod 7$? Ukažme si různé způsoby výpočtu, přičemž nejrychlejší je ten první

1. $3^2 = 9 \equiv 2 \pmod 7$
   $3^4 = (3^2)^2 \equiv 2^2 \equiv 4 \pmod 7$
   $3^8 = (3^4)^2 \equiv 4^2 = 16 \equiv 2 \pmod 7$

2. $3^8 = 3^4 \cdot 3^4 = (81 \pmod 7) \cdot (81 \pmod 7) \equiv 4 \cdot 4 = 16 \equiv 2 \pmod 7$,

3. $3^8 = 6561 \equiv 2 \pmod 7$, protože $6561 = 937 \cdot 7 + 2$ je vhodné se vyhnout násobení velkých čísel (v praxi o velikosti několika set bitů) a provést redukci modulo co nejdříve.

▲

Příklady k procvičení

1. Určete všechny zbytkové třídy modulo 11.
2. Určete aditivní opačný prvek ke každému prvku ze $\mathbb{Z}_8$.
3. Určete multiplikativní inverzní prvek ke každému prvku ze $\mathbb{Z}_8$ (existuje-li).
4. Určete multiplikativní inverzní prvek ke každému prvku ze $\mathbb{Z}_{11}$ (existuje-li).
5. Spočtěte $11^{13} \pmod{53}$.
6. Spočtěte $5^9 \pmod{42}$.

Kapitola 2

Složitost

Bezpečnost mnoha kryptografických algoritmů spočívá ve výpočetní obtížnosti problémů, na kterých je jejich princip založen (problém faktorizace velkého celého čísla, problém diskrétního logaritmu apod.). Pro tyto problémy často neexistují algoritmy s polynomiální časovou složitostí, v nejlepším případě se jedná o složitost subexponenciální. Rovněž jednoduché operace, jakou je např. modulární umocňování z předchozí kapitoly, pro velká čísla (velká řádově stovky bitů) vyžadují realizaci časově efektivními algoritmy. Proto je tato kapitola věnována nezbytným pojmům z teorie složitosti, tak aby později čtenář porozuměl časové složitosti algoritmů uvedených v dalších kapitolách.

Hlavním cílem teorie složitosti je poskytnout mechanismy pro klasifikaci výpočetních problémů podle prostředků potřebných k jejich vyřešení. Měřené prostředky mohou zahrnovat čas, paměťový prostor, počet procesorů, počet registrů atd., ale obvykle se hlavní důraz klade na čas (a někdy i prostor). Důležitým parametrem každé výpočetní metody je její tedy *složitost*, kterou můžeme chápat jako vztah dané metody k daným prostředkům. Složitost tedy dělíme na *složitost časovou* (časovou složitostí rozumíme funkci, která každé množině vstupních dat přiřazuje počet operací vykonaných při výpočtu podle daného algoritmu) a *složitost paměťovou* (definujeme jako závislost paměťových nároků algoritmu na vstupních datech).

2.1 Úvod do teorie složitosti

Libovolnému algoritmu A přiřadíme funkci t, která udává jeho časovou složitost. To znamená, jestliže algoritmus A zpracuje data D a vydá výsledek $A(D)$, udává $t(D)$ počet elementárních operací, které algoritmus A nad daty D vykoná. Tyto operace můžeme ztotožnit s časovými jednotkami, takže na $t(D)$ můžeme pohlížet jako na čas, který algoritmus A potřebuje ke zpracování dat D.

Časovou složitost t je často možné přirozeným způsobem stanovit nejen v závislosti na konkrétních datech D, ale už na základě znalosti jejich rozsahu $|D|$ (stanoveném například v bitech). Potom $t(n) = m$ znamená, že algoritmus A na data D

rozsahu $n = |D|$ spotřebuje m časových jednotek.

Definice 2.1. *Velikost vstupu* je celkový počet bitů potřebných pro reprezentaci vstupu v běžné binární notaci s použitím vhodného kódování. Často je také velikost reprezentována počtem položek vstupu (vstupních údajů).

Příklad 2.2. Počet bitů v binární reprezentaci pozitivního celého čísla n je $1 + + \lfloor log_2 n \rfloor$ bitů. Pro jednoduchost můžeme takovouto velikost vstupu aproximovat $log_2\ n$.
Mějme matici s r řádky, s sloupci a nechť prvkem matice je nenulové celé číslo velikosti nejvýše n. Pak je velikost vstupu $r \cdot s\ log_2\ n$ bitů.
▲

Definice 2.3. *Doba běhu* algoritmu pro daný vstup je počet vykonaných primitivních operací nebo kroků algoritmu.

Často pod pojmem krok rozumíme bitovou operaci. Pro některé algoritmy bude ale výhodnější reprezentovat krokem něco jiného, jako např. porovnání, strojovou instrukci, modulární násobení, atd. Doba běhu algoritmu v nejhorším případě udává, jak nejdéle může trvat výpočet podle algoritmu s libovolnými vstupními daty. Doba běhu algoritmu v průměrném případě má význam doby, kterou můžeme v průměru očekávat při zadání dat velikosti n, její určení je však obtížné.

Předpokládejme nyní, že pět různých algoritmů A_1, A_2, A_3, A_4, A_5 má časovou složitost danou funkcemi

$$\begin{aligned} t_1(n) &= n \\ t_2(n) &= n \log n \\ t_3(n) &= n^2 \\ t_4(n) &= n^3 \\ t_5(n) &= 2^n \end{aligned}$$

Předpokládejme dále, že elementární operace vykonávané algoritmy trvají $1ms$ a spočítejme, jak rozsáhlá data mohou jednotlivé algoritmy zpracovat za sekundu, za minutu a za hodinu, viz tabulka 2.1[1].

I zběžný pohled na tabulku nás přesvědčí, že u algoritmů, jejichž složitost je dána rychle rostoucí funkcí, se při prodlužování doby výpočtu jen pomaleji dosahuje zpracování dat většího rozsahu.

[1]Uvedený logaritmus je základu 2 (tj. $\log_2 n$, resp. $\lg n$). Nicméně nezáleží na tom, jaký má logaritmus základ, protože platí $\log_a n = \frac{\log_b n}{\log_b a} = \frac{1}{\log_b a} \cdot \log_b n$, kde $\frac{1}{\log_b a}$ je konstanta v $O-$notaci zanedbávaná. Proto můžeme použít notaci $\log n$.

algoritmus	složitost	1s	1min	1hod
$t_1(n)$	n (lineární č. složitost)	1000	$6 \cdot 10^4$	$3,6 \cdot 10^6$
$t_2(n)$	$n \log n$ (logaritmická č. s.)	140	4895	$\approx 2,0 \cdot 10^5$
$t_3(n)$	n^2 (polynomiální č. s.)	31	244	1897
$t_4(n)$	n^3 (polynomiální č. s.)	10	39	153
$t_5(n)$	2^n (exponenciální č. s.)	9	15	21

Tab. 2.1 Rozsah zpracovatelných dat

U výpočetních metod s lineární složitostí se například 10-násobné zrychlení výpočtu projeví 10-násobným zvětšením rozsahu zpracovávaných dat, u metod s kvadratickou složitostí se toto zrychlení projeví zhruba 3-násobným zvětšením rozsahu zpracovávaných dat atd. až u algoritmu A s exponenciální složitostí 2^n se 10-násobné zrychlení projeví zvětšením rozsahu dat zhruba o $3,3$. Dosažení rozsahu dat například $n = 50$ u algoritmu A_5 zrychlováním (nebo prodlužováním) výpočtu už vůbec nepřichází v úvahu.

Jedinou schůdnou cestou je nalezení algoritmu s menší časovou složitostí. Jestliže se například podaří nahradit algoritmus složitosti 2^n algoritmem složitosti n^3, otvírá se tím cesta ke zvládnutí většího rozsahu dat v míře, kterou nelze zrychlováním výpočtů suplovat.

Základním kritériem pro určování časové složitosti výpočetních problémů je jejich algoritmická zvládnutelnost. Je třeba si uvědomit, že existují algoritmicky neřešitelné problémy, pro které nemá smysl zkoušet algoritmy konstruovat. Příkladem je problém sestrojení algoritmu, který by o každém algoritmu měl rozhodnout, zda jeho činnost skončí po konečném počtu kroků či nikoliv. Dále existují problémy, pro které byl nalezen exponenciální dolní odhad časové složitosti. Je to například problém rozhodnutí, zda dva regulární výrazy (ve kterých můžeme navíc jako operaci používat druhou mocninu) jsou ekvivalentní.

Definice 2.4. Nechť f je libovolná funkce v oboru přirozených čísel. Říkáme, že problém T má *časovou složitost nejvýše* f, jestliže existuje algoritmus A pro T takový, že složitost všech ostatních algoritmů je menší nebo rovna složitosti algoritmu A. Funkce f se nazývá *horním odhadem* časové složitosti problému T.

Definice 2.5. Říkáme, že problém T má *časovou složitost alespoň* f, jestliže existuje algoritmus A pro T takový, že $t_A(n) \geqq f(n)$ pro všechna n. V tomto případě je f *dolním odhadem* časové složitosti problému T.

Nalézt horní odhad f složitosti problému T tedy znamená najít nějaký algoritmus A pro T se složitostí nejvýše f. Stanovit dolní odhad g složitosti problému T je svou povahou úkol mnohem těžší, neboť je třeba ukázat, že všechny algoritmy A pro T mají složitost aspoň g.

2.2 Složitostní míry

Podaří-li se vyjádřit časovou či paměťovou složitost algoritmu jako funkci rozsahu vstupních dat, pak pro hodnocení efektivity algoritmu je důležité zejména to, jak roste složitost v závislosti na růstu rozsahu vstupních dat. Jinak řečeno, zajímá nás limitní chování složitosti, tzv. *asymptotická složitost*.

Při zkoumání složitosti problémů jsme často nuceni spokojit se s přesností až na multiplikativní konstantu, tj. mluvíme o složitosti *řádově* f. Formálně se pro asymptotické chování funkcí zavádějí následující značení (notace):

Θ-Značení

Pro každou funkci $g(n)$, označíme zápisem $\Theta(g(n))$ množinu funkcí $\Theta(g(n)) = \{f(n) :$ takových, že existují kladné konstanty c_1, c_2 a n_0 tak, že $0 \leqq c_1 g(n) \leqq f(n) \leqq c_2 g(n)$ pro všechna $n \geqq n_0\}$.

Funkce $f(n)$ patří do množiny $\Theta(g(n))$ jestliže existují kladné konstanty c_1 a c_2 takové, že tato funkce nabývá hodnot mezi $c_1 g(n)$ a $c_2 g(n)$. Skutečnost, že $f(n)$ splňuje předcházející vlastnost zapisujeme „$f(n) = \Theta(g(n))$“. Tento zápis znamená $f(n) \in \Theta(g(n))$.

O-Značení

Θ-značení omezuje asymptoticky funkci zdola a shora. Jestliže budeme chtít omezit funkci jen shora použijeme O-značení.

Pro každou funkci $g(n)$, označíme zápisem $O(g(n))$ množinu funkcí $O(g(n)) = \{f(n) :$ takových, že existují kladné konstanty c a n_0 tak, že $0 \leqq f(n) \leqq cg(n)$ pro všechna $n \geqq n_0\}$.

Ω-Značení

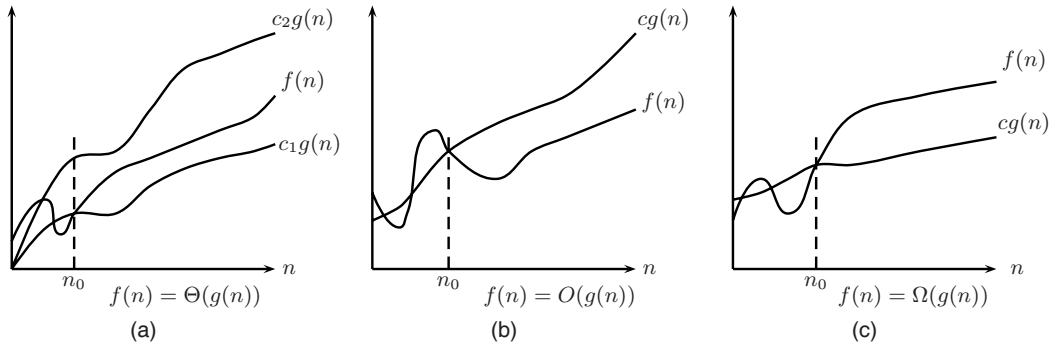
Pro každou funkci $g(n)$, označíme zápisem $\Omega(g(n))$ množinu funkcí $\Omega(g(n)) = \{f(n) :$ takových, že existují kladné konstanty c a n_0 tak, že $0 \leqq cg(n) \leqq f(n)$ pro všechna $n \geqq n_0\}$.

o-Značení

Pro každou funkci $g(n)$, označíme zápisem $o(g(n))$ množinu funkcí $o(g(n)) = \{f(n) :$ takových, že pro každou kladnou konstantu c existuje konstanta n_0 taková, že $0 \leqq f(n) < cg(n)$ pro všechna $n \geqq n_0\}$.

ω-Značení

Pro každou funkci $g(n)$, označíme zápisem $\omega(g(n))$ množinu funkcí $\omega(g(n)) = \{f(n) :$ takových, že pro každou kladnou konstantu c existuje konstanta n_0 taková, že $0 \leqq cg(n) < f(n)$ pro všechna $n \geqq n_0\}$.



Obr. 2.1 Grafické vyjádření Θ, O a Ω značení

n_0 v obr. 2.1 představuje nejmenší možnou hodnotu vyhovující kladeným požadavkům; každá vyšší hodnota samozřejmě také vyhovuje. **(a)** Θ notace ohraničuje funkci mezi dva konstantní faktory. Píšeme, že $f(n) = \Theta(g(n))$, jestliže existují kladné konstanty n_0, c_1 a c_2 takové, že počínaje n_0, hodnoty funkce $f(n)$ vždy leží mezi $c_1g(n)$ a $c_1g(n)$ včetně. **(b)** O značení shora ohraničuje funkci nějakým konstantním faktorem. Píšeme, že $f(n) = O(g(n))$, jestliže existují kladné konstanty n_0 a c takové, že počínaje n_0, hodnoty funkce $f(n)$ jsou vždy menší nebo rovny hodnotě $cg(n)$. **(c)** Ω notace určuje dolní hranici funkce $f(n)$. Píšeme, že $f(n) = \Omega(g(n))$, jestliže existují kladné konstanty n_0 a c takové, že počínaje n_0, hodnoty funkce $f(n)$ jsou vždy větší nebo rovny hodnotě $cg(n)$.

Z pohledu kryptografie je důležité zavést pojem subexponenciální časová složitost, protože některé výpočetně obtížné problémy lze řešit algoritmy, které mají pouze subexponenciální časovou složitost.

Definice 2.6. Algoritmus s *polynomiální časovou složitostí* je algoritmus, pro který má funkce časové složitosti pro nejhorší případ tvar $O(n^k)$, kde n je velikost vstupu a k je konstanta. Jakýkoliv algoritmus, jehož doba běhu nemůže být takto ohraničená, se nazývá algoritmus s *exponenciální časovou složitostí*.

Definice 2.7. Algoritmus s *subexponciální časovou složitostí* je algoritmus, jehož funkce časové složitosti pro nejhorší případ je ve tvaru $e^{O(n)}$, kde n je velikost vstupu.

Algoritmus s subexponciální časovou složitostí je asymptoticky rychlejší než algoritmus s exponenciální časovou složitostí, i když je asymptoticky pomalejší než algoritmus s polynomiální časovou složitostí.

Mnoho vlastností relací mezi reálnými čísly se velmi dobře přenáší na asymptotické porovnání funkcí.

Reflexivita

$f(n) = \Theta(f(n))$
$f(n) = O(f(n))$
$f(n) = \Omega(f(n))$

Symetrie

$f(n) = \Theta(g(n))$ tehdy a jen tehdy, když $g(n) = \Theta(f(n))$

Tranzitivita

$f(n) = \Theta(g(n))$	a	$g(n) = \Theta(h(n))$	implikuje	$f(n) = \Theta(h(n))$
$f(n) = O(g(n))$	a	$g(n) = O(h(n))$	implikuje	$f(n) = O(h(n))$
$f(n) = \Omega(g(n))$	a	$g(n) = \Omega(h(n))$	implikuje	$f(n) = \Omega(h(n))$
$f(n) = o(g(n))$	a	$g(n) = o(h(n))$	implikuje	$f(n) = o(h(n))$
$f(n) = \omega(g(n))$	a	$g(n) = \omega(h(n))$	implikuje	$f(n) = \omega(h(n))$

Transponovaná symetrie

$f(n) = O(g(n))$	tehdy a jen tehdy, když	$g(n) = \Omega(f(n))$
$f(n) = o(g(n))$	tehdy a jen tehdy, když	$g(n) = \omega(f(n))$

Z předcházejících vlastností je zřejmé, že asymptotické značení pro porovnávání funkcí se velmi podobá porovnávání reálných čísel. Tato podobnost se dá vyjádřit následovně:

$f(n) = \Theta(g(n))$	$\approx$	$x = y$
$f(n) = O(g(n))$	$\approx$	$x \leqq y$
$f(n) = \Omega(g(n))$	$\approx$	$x \geqq y$
$f(n) = o(g(n))$	$\approx$	$x < y$
$f(n) = \omega(g(n))$	$\approx$	$x > y$

Trichotomie

Pro každé dvě reálná čísla x a y platí právě jeden z následujících vztahů: $x < y$, $x = y$ nebo $x > y$. To znamená, že každé dvě reálná čísla jsou porovnatelná, ale tato vlastnost neplatí pro asymptotické porovnávání funkcí. To znamená, že existují funkce $f(n)$ a $g(n)$ takové, že neplatí ani jeden ze vztahů $f(n) = O(g(n))$ nebo $f(n) = \Omega(g(n))$. Například funkce n a $n^{1+\sin(n)}$ nemůžeme porovnat pomocí asymptotické notace.

Příklady k procvičení

1. Rozhodněte zda platí:

   a) $3n = O(n)$,

b) $n^2 = O(n)$,

c) $n = O(n^2)$,

d) $\mathrm{e}^n = O(n^4)$,

e) $\sqrt{n} = O(\log n)$,

f) $\log n = O(n)$.

2. Platí $2^{n+1} = O(2^n)$? Platí $2^{2n} = O(2^n)$?

3. Jak dlouho trvá napočítání do 100000. Vyzkoušejte na vašem počítači algoritmus

```
j=0;
for(i=1; i < 100000; i++)
  j++;
```

4. Odpovězte na předcházející otázku s použitím cyklu *while*.

5. Pro každou funkci $f(n)$ a čas t v následující tabulce určete největší n pro které je problém řešitelný v čase t. Předpokládáme, že doba trvání problému o rozsahu n je $f(n)$ mikrosekund.

$f(n)$	1s	1min	1hod	1den	1rok	1století
$\log n$						
$\sqrt{n}$						
n						
$n \cdot \log n$						
n^2						
n^3						
2^n						
$n!$						

6. Pro řešení daného problému máme k dispozici dva algoritmy A_1 a A_2 s časovou složitostí danou funkcemi

$$\begin{aligned} t_1(n) &= n^2 \\ t_2(n) &= n \log n + 10^{10} \end{aligned}$$

Určete pro které rozsahy dat je lepší algoritmus A_1.

7. Určete počet provedených operací C_N (sledovaná operace je dělení) a řádovou složitost $O()$ následujícího algoritmu. Jak jste složitost určili?

```
for(j = 1; j <= N + 1; j++) {
   for(k = 1; k < N; k++) {
     i = N;
     do {
        i = i / 3;
```

```
      } while (i > 1);
    }
}
```

8. Určete počet operací sčítání C_N a řádovou složitost $O()$ následujícího algoritmu.

```
j = 1;
for(i = 1; i <= N ; i++) {
   while(j < i) {
     j = j + 1;
   }
}
```

9. Mějme algoritmus a sledovanou operaci, pro kterou platí, že počet počet provedení této operace C_n je dán pro pro $N > 1$ rekurentním vztahem

$$C_1 = 1; C_N = C_{N-1} + N.$$

Převedením na nerekurzivní vztah dostaneme

$$C_N = \frac{N * (N+1)}{2}.$$

Takový algoritmus má algoritmus řádovou složitost $O(N^2)$.

Určete obdobným způsobem řádovou složitost pro algoritmy s počtem provedení sledované operace C_n, kde:

a) $C_1 = 0; C_N = C_{N/2} + 1$ pro $N > 1$,

b) $C_1 = 0; C_N = C_{N/2} + N$ pro $N > 1$.

Kapitola 3

Algebraické struktury

Tato kapitola se věnuje základním algebraickým strukturám a jejich vlastnostem tak, abychom pochopili design algoritmů, které závisí zejména na vlastnostech určitých algebraických struktur (např. algoritmus AES). Stěžejní algebraickou strukturou je konečné těleso, jehož prvky jsou buď celá čísla modulo prvočíslo nebo prvky tělesa jsou polynomy se specifickými vlastnostmi.

Připomeňme si na začátku, že *binární operace* $\circ$ na množině M je zobrazení $M \times M \longrightarrow M$. Tedy operace $\circ$ je pravidlo, které každé uspořádané dvojici prvků z množiny M přiřazuje opět prvek z množiny M.

3.1 Grupy

Definice 3.1. *Grupou* $(G, \circ)$ nazýváme množinu G (tzv. nosič) spolu s binární operací $\circ$ na množině G, která splňuje následující axiomy:

i) $\forall a, b \in G, a \circ b \in G$

ii) Asociativita: $\forall a, b, c \in G, (a \circ b) \circ c = a \circ (b \circ c)$

iii) Existence neutrálního prvku: $\exists \mathrm{e} \in G, \forall a \in G, a \circ \mathrm{e} = \mathrm{e} \circ a = a$

iv) Existence inverzního prvku: $\forall a \in G, \exists i \in G, a \circ i = i \circ a = \mathrm{e}$

Platí-li navíc následující axiom, hovoříme o *grupě abelovské*.

v) Komutativita: $\forall a, b \in G, a \circ b = b \circ a$

Poznámka 3.2. Dále:

- Pokud pro $(G, \circ)$ platí pouze axiomy i) a ii), nazývá se taková algebraická struktura *pologrupa*.

$+$	0	1	2	3	4	5
0	0	1	2	3	4	5
1	1	2	3	4	5	0
2	2	3	4	5	0	1
3	3	4	5	0	1	2
4	4	5	0	1	2	3
5	5	0	1	2	3	4

$\cdot$	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	1	2	3	4	5
2	0	2	4	0	2	4
3	0	3	0	3	0	3
4	0	4	2	0	4	2
5	0	5	4	3	2	1

a	$(-a)$	a^{-1}
0	0	–
1	5	1
2	4	–
3	3	–
4	2	–
5	1	5

Tab. 3.1 Operace $(+)$, $(\cdot)$ a opačný a inverzní prvek v $\mathbb{Z}_6$

- Pokud pro $(G, \circ)$ platí pouze axiomy i), ii) a iii), označujeme takovou algebraickou strukturu jako *monoid*.
- Pokud operace $(\circ)$ představuje sčítání $(+)$, hovoříme o *aditivní grupě*, neutrální prvek označujeme jako 0 (*nulový prvek*) a inverzní prvek značíme jako $-a$ (opačný prvek).
- Pokud operace $(\circ)$ představuje násobení $(\cdot)$, hovoříme o *multiplikativní grupě*, neutrální prvek označujeme jako 1 (*jednotkový prvek*) a inverzní prvek značíme jako a^{-1}.
- Grupa $(G, \circ)$ je konečná, jestliže počet prvků $|G|$ je konečný. $|G|$ je nazýván *řádem grupy*.
- Umocňování v grupě budeme definovat jako opakovanou aplikaci grupové operace $(\circ)$, např. $a^3 = a \circ a \circ a$.

Příklad 3.3. Množina celých čísel $\mathbb{Z}$ spolu s operací sčítání $(+)$ tvoří abelovskou grupu. Množina $\mathbb{Z}_n$ s operací sčítání modulo n tvoří grupu řádu n. Množina $\mathbb{Z}_n$ s operací násobení modulo n netvoří grupu, protože inverzní prvek neexistuje ke všem prvkům z $\mathbb{Z}_n$.

Nechť $n = 6$. Tabulky v tab. 3.1 představují postupně výsledky operace sčítání modulo 6, dále násobení modulo 6 a v poslední tabulce vidíme opačný a inverzní prvek (pokud existuje) pro prvky ze $\mathbb{Z}_6$. Aditivní opačný prvek k prvku a je $-a \mid a + (-a) \equiv 0 \pmod 6$ a existuje pro každé $a \in Z_6$, např. $2 + 4 \equiv 0 \pmod 6$.

Multiplikativní inverzní prvek k prvku a je $a^{-1} \mid a \cdot a^{-1} \equiv 1 \pmod n$. Takovýto inverzní prvek existuje tehdy a jen tehdy když $NSD(a, n) = 1$, viz kap. 1.3. Např. $5 \cdot 5 \equiv 1 \pmod 6$, ale ke 2 inverzní prvek neexistuje.

▲

Definice 3.4. Nechť $(G, \circ)$ je grupa, $(H, \circ)$ je *podgrupa* grupy $(G, \circ)$, jestliže je $(H, \circ)$ grupa a $H \subseteq G$.

Definice 3.5. Grupa $(G, \circ)$ je *cyklická*, jestliže existuje prvek $a \in G$ takový že pro každé $b \in G$ existuje celé číslo i takové že $b = a^i$. Takový prvek a se nazývá *generátor* grupy G.

Definice 3.6. Nechť $(G, \circ)$ je grupa a $a \in G$. *Řád prvku* a je definován jako nejmenší kladné celé číslo t takové, že $a^t = 1$, pokud takové t existuje. Pokud takové číslo neexistuje, je řád prvku a definován jako ∞.

Poznámka 3.7. Platí:

- Jestliže $(G, \circ)$ je grupa a $a \in G$, potom množina všech mocnin prvku a tvoří cyklickou podgrupu grupy $(G, \circ)$, kterou nazýváme *podgrupa generovaná prvkem* a a značíme $\langle a \rangle$.
- Nechť $(G, \circ)$ je grupa a $a \in G$ je prvek řádu t. Potom $|\langle a \rangle|$, počet prvků podgrupy generované prvkem a, je roven t.
- Každá podgrupa cyklické grupy $(G, \circ)$ je také cyklická. Jestliže je cyklická grupa řádu n, potom pro každý kladný dělitel d čísla n grupa $(G, \circ)$ obsahuje právě jednu podgrupu řádu d.

Věta 3.8 (Lagrange). *$(G, \circ)$ a H její podgrupa, potom $|H|$ dělí $|G|$. Proto jestliže $a \in G$, řád prvku a dělí $|G|$.*

Nyní nechť binární operace $(\circ)$ představuje konkrétně operaci násobení (použijeme symbol $*$).

Definice 3.9. *Multiplikativní grupa množiny* $\mathbb{Z}_n$ je $\mathbb{Z}_n^* = \{a \in \mathbb{Z}_n \mid NSD(a, n) = 1\}$. Jestliže n je prvočíslo, pak $\mathbb{Z}_n^* = \{a \mid 1 \leqq a \leqq n - 1\}$.

Poznámka 3.10. Dále:

- Nechť $a \in \mathbb{Z}_n^*$. Řád prvku a je nejmenší kladné celé číslo t takové, že $a^t \equiv 1 \pmod{n}$
- Nechť $a \in \mathbb{Z}_n^*$. Jestliže je řád prvku a roven $\phi(n)$, nazýváme jej *generátorem* (někdy také primitivním prvkem) grupy $\mathbb{Z}_n^*$. Pokud má grupa $\mathbb{Z}_n^*$ generátor, potom je to grupa cyklická.

Příklad 3.11. Nechť $n = 21$. Pak $\mathbb{Z}_{21}^* = \{1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20\}$. Platí $\phi(21) = \phi(7) * \phi(3) = 12 = |\mathbb{Z}_{21}^*|$. Řády jednotlivých prvků z $\mathbb{Z}_{21}^*$ vidíme v tabulce 3.2. Grupa $\mathbb{Z}_{21}^*$ není cyklická, protože neobsahuje prvek řádu $\phi(n) = 12$. Např. $\mathbb{Z}_5^*$ je cyklická, protože má generátor $a = 2$ (ověřte).
▲

$a \in \mathbb{Z}_n^*$	1	2	4	5	8	10	11	13	16	17	19	20
řád a a	1	6	3	6	2	6	6	2	3	6	6	2

Tab. 3.2 Řády prvků z $\mathbb{Z}_{21}^*$

Příklad 3.12. Mějme multiplikativní grupu $\mathbb{Z}_{19}^* = \{1, 2, \ldots, 18\}$ řádu 18. Grupa je cyklická a její generátor $a = 2$. V tabulce 3.3 lze nalézt podgrupy grupy $\mathbb{Z}_{19}$ a jejich generátory.

podgrupa	generátor	řád
$\{1\}$	1	1
$\{1, 18\}$	18	2
$\{1, 7, 11\}$	$7, 11$	3
$\{1, 7, 8, 11, 12, 18\}$	$8, 12$	6
$\{1, 4, 5, 6, 7, 9, 11, 16, 17\}$	$4, 5, 6, 9, 16, 17$	9
$\{1, 2, \ldots, 18\}$	$2, 3, 10, 13, 14, 15$	18

Tab. 3.3 Podgrupy grupy $\mathbb{Z}_{19}^*$

▲

Příklady k procvičení

1. Rozhodněte, zda $(\mathbb{N}, +), (\mathbb{N}, \cdot), (\mathbb{Z}, \cdot), (\mathbb{R}, +), (\mathbb{R}, \cdot)$ tvoří pologrupu nebo (abelovskou) grupu.

2. Rozhodněte, zda $H = \{x \in \mathbb{Z}_{26}$, kde $13x \equiv 0 \pmod{26}\}$, je podgrupa grupy $(\mathbb{Z}_{26}, +)$.

3. Rozhodněte, zda $H = \{x \in \mathbb{Z}_5$, kde $x^2 \equiv 1 \pmod{5}\}$, je podgrupa grupy $(\mathbb{Z}_5 \setminus \{0\}, \cdot)$.

4. Mějme pravidelný šestiúhelník s vrcholy $a, b, c, d, \mathrm{e}, f$. Mějme operaci rotace šestiúhelníku po směru chodu hodinových ručiček o násobky 60°, tj. $r_0 : 0°, r_1 : 60°, r_2 : 120°$, $r_3 : 180°, r_4 : 240°, r_5 : 300°$. Nechť $R = \{r_0, r_1, r_2, r_3, r_4, r_5\}$. Definujme operaci na prvcích množiny R takto: $r_i \circ r_j$ je rotace získaná provedením nejprve rotace r_i a poté rotace r_j.

   a) Ukažte, že R spolu s operací $(\circ)$ tvoří grupu.

   b) Je $(R, \circ)$ abelovská grupa?

   c) Najděte všechny prvky množiny R, pro které $r \circ r \circ r = \mathrm{e}$.

5. Je grupa $\mathbb{Z}_{11}^*$ cyklická? Jaký je její řád? Jaké má generátory? Jaké má podgrupy, jaké jsou jejich řády a generátory?

6. Je grupa $\mathbb{Z}_{10}^*$ cyklická? Jaký je její řád? Jaké má generátory? Jaké má podgrupy, jaké jsou jejich řády a generátory?

3.2 Tělesa

Definice 3.13. *Okruhem* $(R, +, \cdot)$ nazýváme množinu R (R z anglického ring = okruh) spolu se dvěmi binárními operacemi $(+)$ (sčítání) a $(\cdot)$ (násobení) splňující následující axiomy:

i) $(R, +)$ je abelovská grupa s nulovým prvkem.

ii) $(R, \cdot)$ je pologrupa.

iii) Operace $(\cdot)$ je distributivní vůči $(+)$, tj. $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$ a $(a + b) \cdot c = (a \cdot c) + (b \cdot c)$ $\forall a, b, c \in R$.

iv) Okruh je *komutativním okruhem* jestliže $\forall a, b \in R, a \cdot b = b \cdot a$.

v) Okruh nazýváme okruhem s jednotkou *okruh s jednotkou* má-li pologrupa $(R, \cdot)$ neutrální prvek.

Definice 3.14. *Tělesem* $(F, +, \cdot)$ nazýváme komutativní okruh s jednotkou (F z anglického field (těleso = field), občas se také můžeme setkat místo tělesa s pojmem pole), ve kterém platí navíc axiom:

v) Existence multiplikativního inverzního prvku:
$\forall a \in F, a \neq 0, \exists a^{-1} \in F, a \cdot a^{-1} = a^{-1} \cdot a = 1$.

Příklad 3.15. Následují příklady okruhů a těles:

- Množina celých čísel tvoří komutativní okruh.
- Množina celých čísel netvoří těleso, protože jediné dva nenulové prvky, které mají multiplikativní inverzní prvek jsou 1 a -1.
- Množina reálných čísel $\mathbb{R}$, racionálních čísel $\mathbb{Q}$ a komplexních čísel $\mathbb{C}$ tvoří těleso.
- Mějme množinu celých čísel modulo n, pro $n > 1$. Jestliže čísla $a \in \mathbb{Z}_n$ a n mají společné dělitele, pak po násobení čísel z množiny $\{0, 1, \ldots, n-1\}$ číslem a nedostaneme kompletní množinu zbytků (např. pro $a = 6, n = 8$ ne (čtenář nechť se přesvědčí sám), ale pro $a = 5, n = 8$ celou množinu zbytků dostaneme).

  Tj. pokud se více než jedno číslo z množiny $\mathbb{Z}_n = \{0, 1, 2, .., n-1\}$ mapuje na stejný zbytek po dělení, neexistuje unikátní inverzní prvek vzhledem k násobení a není splněn axiom v) předchozí definice.

▲

Poznámka 3.16. Množina celých čísel s operacemi $(+)$ a $(\cdot)$ modulo n, pro $n > 1$ tvoří těleso jen tehdy, když n je prvočíslo:

- Sčítání je uzavřené, tj. pro každé $a, b \in Z_n$ platí $(a + b) \pmod{n} \in Z_n$.
- Sčítání je asociativní, tj. pro každé $a, b, c \in Z_n$ platí

$$[(a + b) + c] \pmod{n} = [a + (b + c)] \pmod{n}.$$

- Aditivní jednotkový prvek 0:

$$(a + 0) \pmod{n} = (0 + a) \pmod{n} = a \pmod{n}$$

  pro každé $a \in Z_n$.

- Sčítání je komutativní, tj. pro každé $a, b \in Z_n$ platí

$$(a + b) \pmod{n} = (b + a) \pmod{n}.$$

- Aditivní opačný prvek k prvku a je $-a$:

$$a + (-a) \equiv 0 \pmod{n}$$

  tj. $-a = n - a$ pro každé $a \in Z_n$.

- Násobení je uzavřené, tj. pro každé $a, b \in Z_n$ platí $(a \cdot b) \pmod{n} \in Z_n$.
- Násobení je asociativní, tj. pro každé $a, b, c \in Z_n$ platí

$$[(a \cdot b) \cdot c] \pmod{n} = [a \cdot (b \cdot c)] \pmod{n}.$$

- Násobení je komutativní, tj. pro každé $a, b \in Z_n$ platí

$$(a \cdot b) \pmod{n} = (b \cdot a) \pmod{n}.$$

- Multiplikativní jednotkový prvek 1:

$$(a \cdot 1) \pmod{n} = (1 \cdot a) \pmod{n} = a \pmod{n}$$

  pro každé $a \in Z_n$.

- Multiplikativní inverzní prvek k prvku a je a^{-1}: $a \cdot a^{-1} \equiv 1 \pmod{n}$. Takovýto inverzní prvek existuje tehdy a jen tehdy když $NSD(a, n) = 1$.
- Platí distributivita vzhledem ke sčítání.

3.3 Konečná tělesa

V kryptografických algoritmech hrají důležitou roli konečná tělesa, nikoli nekonečná jako je např. množina reálných čísel s obvyklými operacemi sčítání a násobení. Příkladem jsou aplikace konečných těles je kryptografie na bázi eliptických křivek (viz kapitola 7), schémata pro sdílení klíčů, S-boxy blokových šifer, symetrický šifrovací algoritmus AES atd.

Definice 3.17. *Konečné těleso* je těleso, kde množina F obsahuje konečný počet prvků. Počet prvků množiny F nazýváme *řád konečného tělesa* F.

Lze dokázat, že řád konečného tělesa musí být m-tou mocninou prvočísla p, kde je $m \geqq 1$. Konečná tělesa řádu p^m se často označují jako $GF(p^m)$ z anglického Galois Fields - *Galoisova tělesa* - podle Évarista Galoise, zakladatele moderní abstraktní algebry. Toto označení budeme používat také v našich skriptech.

3.3.1 Konečná tělesa $GF(p)$

Z $GF(p^m)$ pro $m = 1$ dostáváme konečné těleso $GF(p)$, což není nic jiného než množina $\mathbb{Z}_p$ celých čísel $\{0, 1, \ldots, p-1\}$ spolu s aritmetickými operacemi modulo prvočíslo p, viz příklad 3.15.

Příklad 3.18. Příklady $GF(p)$:

- Jako příklad uveďme (viz tabulka 3.4) konečné těleso $GF(5)$:

$+$	0	1	2	3	4
0	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	1
3	3	4	0	1	2
4	4	0	1	2	3

$\cdot$	0	1	2	3	4
0	0	0	0	0	0
1	0	1	2	3	4
2	0	2	4	1	3
3	0	3	1	4	2
4	0	4	3	2	1

a	$(-a)$	a^{-1}
0	0	–
1	4	1
2	3	3
3	2	2
4	1	4

Tab. 3.4 Operace $(+)$, $(\cdot)$ a opačný a inverzní prvek v $GF(5)$

- Nejjednodušším příkladem je těleso $GF(2)$ (viz tabulka 3.5), kde $Z_2 = \{0, 1\}$ a výsledky operací $(+)$ a $(\cdot)$ jsou evidentní. Operace sčítání je ekvivalentní operaci XOR a operace násobení operaci AND.

▲

$+$	0	1
0	0	1
1	1	0

$\cdot$	0	1
0	0	0
1	0	1

a	$(-a)$	a^{-1}
0	0	–
1	1	1

Tab. 3.5 Operace $(+)$, $(\cdot)$ a opačný a inverzní prvek v $GF(2)$

3.3.2 Konečná tělesa $GF(2^m)$

Většina šifrovacích algoritmů vyžaduje operace s celými čísly. Vzhledem k bitové reprezentaci celých čísel při praktické implementaci algoritmů často vyžadujeme celá čísla, která se vejdou do daného počtu bitů. A to jsou celá čísla, která se vejdou do m-bitového slova, tj. čísla, která jsou v rozsahu $0, \ldots, 2^m - 1$.

Proto je mezi tělesy $GF(p^m), m > 1$ pozornost upřena zejména na tělesa pro $p = 2$, tj. $GF(2^m), m > 1$. Jejich prvky lze reprezentovat m-bitovými slovy. Např. pro $m = 8$ máme 8 bitové slovo, kterým můžeme reprezentovat hodnoty $0, \ldots, 2^8 - 1$ tj. hodnoty $0, \ldots, 255$ (pozor, např. $\mathbb{Z}_{256}$ není těleso, ale $GF(2^8)$ těleso je - ověřte). Výsledky operací v $GF(2^3)$ vidíme v tabulce 3.6. Přestože nám na první pohled mohou připadat podivné, po prostudování zbytku kapitoly by nám měly být tyto výsledky zřejmé.

$+$	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	0	3	2	5	4	7	6
2	2	3	0	1	6	7	4	5
3	3	2	1	0	7	6	5	4
4	4	5	6	7	0	1	2	3
5	5	4	7	6	1	0	3	2
6	6	7	4	5	2	3	0	1
7	7	6	5	4	3	2	1	0

$\cdot$	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	3	1	7	5
3	0	3	6	5	7	4	1	2
4	0	4	3	7	6	2	5	1
5	0	5	1	4	2	7	3	6
6	0	6	7	1	5	3	2	4
7	0	7	5	2	1	6	4	3

a	$(-a)$	a^{-1}
0	0	–
1	1	1
2	2	5
3	3	6
4	4	7
5	5	2
6	6	3
7	7	4

Tab. 3.6 Operace $(+)$, $(\cdot)$ a opačný a inverzní prvek v $GF(2^3)$

Počítání s polynomy

Abychom si mohli vysvětlit tabulku 3.6, přejdeme nyní k *polynomiální aritmetice* a ukážeme si obvyklé aritmetické operace s polynomy. *Polynom* (v proměnné x) stupně m vyjádříme ve tvaru

$$f(x) = a_m x^m + a_{m-1} x^{m-1} + \ldots + a_1 x^1 + a_0 = \sum_{i=0}^{m} a_i x^i,$$

kde koeficienty polynomu a_i jsou prvky z určené množiny hodnot. Stupeň polynomu je největší takové $m \in \mathbb{N}$ takové, že $a_m \neq 0$.

Součet dvou polynomů $f(x) = \sum_{i=0}^{n} a_i x^i$ a $g(x) = \sum_{i=0}^{m} b_i x^i$, kde $n \geqq m$, je definován

$$f(x) + g(x) = \sum_{i=0}^{n} (a_i + b_i) x^i + \sum_{i=m+1}^{n} a_i x^i.$$

Násobení polynomů $f(x)$ a $g(x)$ je definován jako

$$f(x) \cdot g(x) = \sum_{i=0}^{m+n} c_i x^i,$$

kde $c_k = a_0 b_k + a_1 b^{k-1} + \ldots + a_{k-1} b^1 + a_k b^0$ a $a_i = 0$ pro $i > n$ a $b_i = 0$ pro $i > m$.

Dělení dvou polynomů je poněkud komplikovanější, protože je možné jen tehdy, když připustíme existenci zbytku po dělení:

$$\tfrac{f(x)}{g(x)} = q(x) + \tfrac{r(x)}{g(x)}, \text{ tj. } f(x) = q(x)g(x) + r(x),$$

kde polynom $f(x)$ je stupně n, polynom $g(x)$ je stupně m, $n \geqq m$ a polynom $r(x)$ resp. $q(x)$ představují zbytek po dělení resp. podíl. Např. (ověřte)

$$f(x) = q(x)g(x) + r(x) : \ (x^3 + x^2 + 2) = (x + 2)(x^2 - x + 1) + x.$$

Příklad 3.19. Příklady polynomiální aritmetiky následují.

- Sčítání:

$$\begin{array}{rrrr} (x^3 & +x^2 & -x & +2) \\ + & (x^2 & +x & +1) \\ \hline x^3 & +2x^2 & & +3 \end{array}$$

- Odčítání:

$$\begin{array}{rrrr} (x^3 & +x^2 & -x & +2) \\ - & (x^2 & +x & +1) \\ \hline x^3 & & -2x & -1 \end{array}$$

- Násobení:

$$\begin{array}{rrrrrr} & & (x^3 & +x^2 & -x & +2) \\ & & . & (x^2 & +x & +1) \\ \hline & & (x^3 & +x^2 & -x & +2) \\ & x^4 & +x^3 & -x^2 & +2x & \\ x^5 & +x^4 & -x^3 & +2x^2 & & \\ \hline x^5 & +2x^4 & +x^3 & +2x^2 & +x & +2 \end{array}$$

- Dělení:

$$\begin{array}{rrrrcrrrl}
(x^3 & +x^2 & -x & +2) & / & (x^2 & -x & +1) & = x+2 \\
\cline{1-4}\cline{6-8}
-(x^3 & -x^2 & +x &) & & & & & \\
\cline{1-4}
 & (2x^2 & -2x & +2) & & & & & \\
- & (2x^2 & -2x & +2) & & & & & \\
\cline{1-4}
 & & & 0 & & & & &
\end{array}$$

▲

Pokud jsou však polynomy prvky konečného tělesa $GF(p^m)$, musí být výsledkem operace opět prvek (polynom) z tohoto tělesa. To znamená, že ani koeficienty polynomu, ani stupeň polynomu nemohou mít libovolnou hodnotu.

Mějme nyní množinu M polynomů stupně nejvýše $m-1$ nad tělesem $\mathbb{Z}_p$. Každý polynom má tvar

$$f(x) = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \ldots + a_1x^1 + a_0 = \sum_{i=0}^{m-1} a_i x^i,$$

kde koeficienty $a_i \in \mathbb{Z}_p = \{0, 1, \ldots, p-1\}$. Existuje p^m různých polynomů v množině M, viz příklad 3.21. Množina M spolu s odpovídajícími aritmetickými operacemi tvoří těleso. Operace s koeficienty jsou prováděny modulo p, tj. platí pro ně operace jako pro $\mathbb{Z}_p$.

Pokud je výsledkem některé operace polynom $f(x)$ stupně většího než $m-1$, musíme provést jeho redukci modulo ireducibilní polynom $m(x)$ stupně m. Ireducibilní polynom je analogií prvočísla, nelze ho faktorizovat, viz definice 3.20. Polynom $f(x)$ stupně většího než $m-1$ pak vydělíme polynomem $m(x)$ a získáme polynom $r(x)$ reprezentující zbytek po dělení, tj.

$$r(x) = f(x) \pmod{m(x)}.$$

Definice 3.20. Nechť $f(x) \in M$ je polynom stupně > 1. Polynom se nazývá *ireducibilní* nad $\mathbb{Z}_p$ pokud jej nemůžeme vyjádřit jako součin dvou polynomů z M stupně nižšího než má polynom $f(x)$.

Už víme, že pro kryptografii jsou zajímavá tělesa $GF(2^m), m > 1$. Polynom $f(x)$ v $GF(2^m), m > 1$,

$$f(x) = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + \ldots + a_1x^1 + a_0 = \sum_{i=0}^{m-1} a_i x^i,$$

má množinu koeficientů $\mathbb{Z}_2$, tj. $a_i \in \{0, 1\}$. Proto může být každý polynom z $GF(2^m)$ jednoznačně reprezentován m-ticí bitů $\{a_{m-1}, a_{n-2}, \ldots, a_1, a_0\}$, tedy jako m-bitové číslo. Operace $(+)$ a $(\cdot)$ mohou být realizovány pomocí bitových operací (což v konečném důsledku přináší výhodu rychlosti takto realizovaných operací).

Příklad 3.21. Jako příklad tělesa $GF(2^m), m > 1$ uveďme těleso $GF(2^3)$ a ukažme si operace s polynomy (i bitově). Operace v $GF(2^3)$ se provádějí modulo ireducibilní polynom stupně 3, ty jsou dva: $(x^3 + x + 1)$ a $(x^3 + x^2 + 1)$. Těleso má celkem 8 prvků, jsou to polynomy $0, 1, x, x+1, x^2, x^2+1, x^2+x, x^2+x+1$. Připomeňme ještě jednou, že operace s koeficienty jsou prováděny modulo 2.

Sčítání polynomů v $GF(2^3)$ (výsledkem součtu dvou polynomů stupně $\leqq 2$ bude polynom opět stupně $\leqq 2$) bude realizováno takto:

$$\begin{array}{rrrr} & & (x & +1) \\ + & (x^2 & &) \\ \hline & x^2 & +x & +1 \end{array} \qquad \begin{array}{rr} & 011 \\ \oplus & 100 \\ \hline & 111 \end{array} \qquad \begin{array}{rrrr} & & (x & +1) \\ + & (x^2 & & +1) \\ \hline & (x^2 & +x &) \end{array} \qquad \begin{array}{rr} & 011 \\ \oplus & 101 \\ \hline & 110 \end{array}$$

▲

Příklad 3.22. Násobení polynomů modulo ireducibilní polynom $m(x) = (x^3+x+1)$ pro polynomy $(x+1)$ a $(x+1)$ provedeme takto (symbol $\ll$ představuje bitový posun doleva):

$$\begin{array}{lll} (x+1)\cdot(x+1) & = (x+1)\cdot x + (x+1)\cdot 1 & = x^2+1 \\ 011 \cdot 011 & = 011 \ll 1 \oplus 011 \ll 0 \quad = 110 \oplus 011 & = 101 \end{array}$$

V prvním příkladě nebylo potřeba provádět redukci modulo $m(x)$, protože výsledným součinem je polynom stupně $m \leqq 2$. Součin v následujícím příkladu však bude polynom stupně $> m-1$, tj. > 2, a bude třeba provést jeho redukci modulo ireducibilní polynom $m(x) = (x^3 + x + 1)$:

$$\begin{array}{lll} (x^2+1)\cdot(x^2+x) & = (x^2+x)\cdot x^2 + (x^2+x)\cdot 1 & = x^4+x^3+x^2+x \\ 101 \cdot 110 & = 110 \ll 2 \oplus 110 \ll 0 = 11000 \oplus 110 & = 11110 \end{array}$$

Součin, polynom stupně 4, musíme „redukovat“, redukci modulo ireducibilní polynom vyjádříme nejprve pomocí polynomů:

$$\begin{array}{rrrrr} (x^4 & +x^3 & +x^2 & +x &) \\ (x^4 & & +x^2 & +x &) \\ \hline (& x^3 & & &) \\ (& x^3 & & +x & +1) \\ \hline & & & (x & +1) = r(x) \end{array} \qquad (\mathrm{mod}\ (x^3 \quad +x \quad +1)) \quad = (x+1) = q(x)$$

Nyní ukážeme redukci modulo ireducibilní polynom bitově:

$$\begin{array}{ll} 11110\ (\mathrm{mod}\ 1011) = 11110 \oplus 1011 \ll 1 & = 1000\ (\mathrm{mod}\ 1011) = 1000 \oplus 1011 = 011 \\ (x^4+x^3+x^2+x)\ (\mathrm{mod}\ (x^3+x+1)) & = x^3\ (\mathrm{mod}\ (x^3+x+1)) = (x+1) \end{array}$$

Výsledkem násobení polynomů $(x^2+1)\cdot(x^2+x)$ je polynom $r(x) = (x+1) = (x^4 + + x^3 + x^2 + x) \pmod{(x^3+x+1)} = f(x) \pmod{m(x)}$.

Celý příklad ale můžeme vyřešit méně komplikovaně, pokud si uvědomíme, jak funguje násobení nějakého polynomu z $GF(2^3)$ mocninami polynomu x (modulo (x^3+x+1)). Pro náš polynom (x^2+1) dostaneme:

- $(x^2+1)\cdot x = (101) \ll 1 = (101)\cdot(010) = (1010) \pmod{1011} = (1010)\oplus(1011) = = (001) = 1$
- $(x^2+1)\cdot x^2 = (101) \ll 2 = (101)\cdot(100) = (10100) \pmod{1011} = = (10100)\oplus(1011) \ll 1 = (10100)\oplus(10110) = (010) = x$

Symbol $\ll$ opět představuje bitový posun doleva. Celé násobení polynomů

$$(x^2+1)\cdot(x^2+x) = (x+1)$$

lze bitově zapsat takto:

$$(101)\cdot(110) = (101)\cdot[(100)\oplus(010)] = (010)\oplus(001) = (011).$$

▲

Z tabulek 3.7 a 3.8, které obsahují výsledky operací v tělese $GF(2^3)$ polynomiálně i bitově, už jsou hodnoty v tab. 3.6 zcela zřejmé.

Příklady k procvičení

1. Nalezněte všechny prvky tělesa $GF(2^4)$.
2. Spočtěte obvyklým způsobem součet a součin následujících polynomů (polynomy nejsou prvkem žádného speciálního tělesa):
   - $(x^2+x+1)+(x^4+x^3+x^2+1)$
   - $(x^3+x^2+x)+(x^3+x+1)$
   - $(x^4+x+1)+(x^4+x^3+x^2+x+1)$
   - $(x^2+x+1)\cdot(x^4+x^3+x^2+1)$
   - $(x^3+x^2+x)\cdot(x^3+x+1)$
   - $(x^4+x+1)\cdot(x^4+x^3+x^2+x+1)$
3. Vyřešte v $GF(2^4)$ modulo ireducibilní polynom $m(x) = x^4+x+1$:
   - $(x^2+1)+(x^3+x^2+1)$
   - $(x^2+1)+(x+1)$
   - $(x^3+x^2+x)+(x^3+x+1)$

$+$	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
0	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
1	1	0	$x+1$	x	x^2+1	x^2	x^2+x+1	x^2+x
x	x	$x+1$	0	1	x^2+x	x^2+x+1	x^2	x^2+1
$x+1$	$x+1$	x	1	0	x^2+x+1	x^2+x	x^2+1	x^2
x^2	x^2	x^2+1	x^2+x	x^2+x+1	0	1	x	$x+1$
x^2+1	x^2+1	x^2	x^2+x+1	x^2+x	1	0	$x+1$	x
x^2+x	x^2+x	x^2+x+1	x^2	x^2+1	x	$x+1$	0	1
x^2+x+1	x^2+x+1	x^2+x	x^2+1	x^2	$x+1$	x	1	0

$\cdot$	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
0	0	0	0	0	0	0	0	0
1	0	1	x	$x+1$	x^2	x^2+1	x^2+x	x^2+x+1
x	0	x	x^2	x^2+x	$x+1$	1	x^2+x+1	x^2+1
$x+1$	0	$x+1$	x^2+x	x^2+1	x^2+x+1	x^2	1	x
x^2	0	x^2	$x+1$	x^2+x+1	x^2+x	x	x^2+1	1
x^2+1	0	x^2+1	1	x^2	x	x^2+x+1	$x+1$	x^2+x
x^2+x	0	x^2+x	x^2+x+1	1	x^2+1	$x+1$	x	x^2
x^2+x+1	0	x^2+x+1	x^2+1	x	1	x^2+x	x^2	$x+1$

Tab. 3.7 Operace $(+)$, $(\cdot)$ modulo (x^3+x+1) v $GF(2^3)$

$+$	000	001	010	011	100	101	110	111
000	000	001	010	011	100	101	110	111
001	001	000	011	010	101	100	111	110
010	010	011	000	001	110	111	100	101
011	011	010	001	000	111	110	101	100
100	100	101	110	111	000	001	010	011
101	101	100	111	110	001	000	011	010
110	110	111	100	101	010	011	000	001
111	111	110	101	100	011	010	001	000

$\cdot$	000	001	010	011	100	101	110	111
000	000	000	000	000	000	000	000	000
001	000	001	010	011	100	101	110	111
010	000	010	100	110	011	001	111	101
011	000	011	110	101	111	100	001	010
100	000	100	011	111	110	010	101	001
101	000	101	001	100	010	111	011	110
110	000	110	111	001	101	011	010	100
111	000	111	101	010	001	110	100	011

Tab. 3.8 Operace $(+)$, $(\cdot)$ modulo $(x^3 + x + 1)$ v $GF(2^3)$ v bitové reprezentaci

- $(x^3 + x^2 + x) \cdot (x^3 + x + 1)$
- $(x^2 + 1) \cdot (x^3 + x^2 + 1)$
- $(x^2 + 1) \cdot (x + 1)$

4. Vyřešte v $GF(2^5)$ modulo ireducibilní polynom $m(x) = x^5 + x^2 + 1$:
   - $(x^2 + x + 1) + (x^4 + x^3 + x^2 + 1)$
   - $(x^3 + x^2 + x) + (x^4 + x + 1)$
   - $(x^4 + x + 1) + (x^4 + x^3 + x^2 + x + 1)$
   - $(x^2 + 1) + (x^3 + x^2 + 1)$
   - $(x^3 + 1) + (x^2 + 1)$
   - $(x^4 + 1) + (x^3 + x^2 + 1)$
   - $(x^2 + x + 1) \cdot (x^4 + x^3 + x^2 + 1)$
   - $(x^3 + x^2 + x) \cdot (x^4 + x + 1)$
   - $(x^4 + x + 1) \cdot (x^4 + x^3 + x^2 + x + 1)$
   - $(x^2 + 1) \cdot (x^3 + x^2 + 1)$
   - $(x^3 + 1) \cdot (x^2 + 1)$
   - $(x^4 + 1) \cdot (x^3 + x^2 + 1)$

Kapitola 4

Algoritmy

Mnoho kryptografických algoritmů využívá stejné dílčí výpočty např. pro určení největšího společného dělitele dvou c.č., pro určení multiplikativního inverzního prvku, pro rychlé modulární umocňování nebo např. pro testování prvočíselnosti. Vzhledem k již zmiňovanému faktu, že se v kryptografii pracuje s velkými čísly, a vzhledem k nutnosti efektivních a rychlých (hardwarových i softwarových) implementací kryptografických algoritmů, musí být efektivní také algoritmy pro tyto dílčí výpočty. Tato kapitola se proto věnuje některým z nich.

Mějme dvě celá čísla a, b taková, že $0 < a \leqq n, 0 < b \leqq n$. V příkladu 2.2 jsme si ukázali, že počet bitů v binární reprezentaci pozitivního celého čísla je $log_2\ n$. Řádová složitost aritmetických operací s celými čísly je uvedena v tabulce 4.1, pro násobení a dělení ale existují rychlejší algoritmy.

Operace		Bitová složitost
Sčítání	$a + b$	$O(log_2\ a + log_2\ b) = O(log_2\ n)$
Odčítání	$a - b$	$O(log_2\ a - log_2\ b) = O(log_2\ n)$
Násobení	$a \cdot b$	$O((log_2\ a) \cdot (log_2\ b)) = O((log_2\ n)^2)$
Dělení	$a = qb + r$	$O((log_2\ q) \cdot (log_2\ b)) = O((log_2\ n)^2)$

Tab. 4.1 Bitová složitost operací v $\mathbb{Z}$

4.1 Euklidův algoritmus

Euklidův algoritmus je postup, jak zjistit největšího společného dělitele dvou celých čísel pomocí zbytků po celočíselném dělení. Nalezení NSD dvou kladných celých čísel pomocí kanonického rozkladu není vzhledem k použití prvočíselné faktorizace postup právě efektivní.

Euklidův algoritmus (ozn. EA) vychází z faktu, že pro dvě kladná celá čísla $a, b, a > b$ platí, že $\text{NSD}(a, b) = \text{NSD}(b, a \pmod b)$. Např. opakovaně dokud $b \neq$ $\neq 0$ můžeme provádět $\text{NSD}(55, 22) = \text{NSD}(22, 55 \pmod{22}) = \text{NSD}(22, 11) = \text{NSD}(11, 22 \pmod{11}) = \text{NSD}(11, 0) = 11$.

Předpokládejme, že $d = \text{NSD}(a, b)$. Pro každé kladné b můžeme a vyjádřit jako

$$a = q \cdot b + r \equiv r \pmod b,$$
$$a \pmod b = r.$$

Proto $a \pmod b = a - q \cdot b$ pro nějaké c.č. q.

Slovní formulace Euklidova algoritmu:
Nechť a_0 a a_1 jsou dvě kladná celá čísla, $a_0 > a_1$. Známe-li a_{i-1} a a_i, spočítáme $a_{i+1} =$ $= a_{i-1} \pmod{a_i}$. Tedy víme, že existuje takové $q_i \in \mathbb{Z}, q_i > 0$ že $a_{i-1} = q_i \cdot a_i + a_{i+1}$ a $a_{i+1} < a_i$. Algoritmus skončí, když $a_{n+1} = 0$, potom $a_n = \text{NSD}(a_0, a_1)$.

Příklad 4.1. Najděte $\text{NSD}(98, 56)$.

- Zápis pomocí $a_{i+1} = a_{i-1} \pmod{a_i}$:

a_0	$a_0 = 98$
a_1	$a_1 = 56$
$a_2 = a_0 \pmod{a_1}$	$a_2 = 98 \pmod{56} = 42$
$a_3 = a_1 \pmod{a_2}$	$a_3 = 56 \pmod{42} = \mathbf{14} = \text{NSD}(98, 56)$
$a_4 = a_2 \pmod{a_3}$	$a_4 = 42 \pmod{14} = 0$

- Zápis pomocí $a_{i-1} = q_i \cdot a_i + a_{i+1}$:

$a_0 = q_1 \cdot a_1 + a_2$	$98 = 1 \cdot 56 + 42$
$a_1 = q_2 \cdot a_2 + a_3$	$56 = 1 \cdot 42 + \mathbf{14} = \text{NSD}(98, 56)$
$a_2 = q_3 \cdot a_3 + a_4$	$42 = 3 \cdot 14 + 0$

- A ještě jednou a jinak, pomocí $a_i = a_{i-1} - q_i \cdot a_i$:

a_0	$a_0 = 98$
a_1	$a_1 = 56$
$a_2 = a_0 - q_1 \cdot a_1$	$a_2 = 98 - 1 \cdot 56 = 42$
$a_3 = a_1 - q_2 \cdot a_2$	$a_3 = 56 - 1 \cdot 42 = \mathbf{14} = \text{NSD}(98, 56)$
$a_4 = a_2 - q_3 \cdot a_3$	$a_4 = 42 - 3 \cdot 14 = 0$

▲

Uveďme nyní nejčastější verzi EA, který má složitost $O((log_2\ n)^2)$ bitových operací. Víme, že $\text{NSD}(a, b) = \text{NSD}(b, a \pmod b)$ pro $a > b$, a že zbytek r dělí a, b. Také platí $a = q \cdot b + r$, pro nějaké c.č. q.

Vstup: dvě kladná celá čísla a, b kde $a > b$.
Výstup: $\text{NSD}(a, b)$.

```
while(b != 0)
{
    r = a mod b;
    a = b;
    b = r;
}
return (a);
```

4.2 Rozšířený Euklidův algoritmus

Euklidův algoritmus vytváří ze dvou zadaných čísel $a_0 > a_1$ (ostře) klesající posloupnost $(a_i, i = 0, 1, \ldots, n+1)$, jejíž poslední člen $a_{n+1} = 0$ a předposlední $a_n = \mathrm{NSD}(a_0, a_1)$. Přitom $(i+2)$-hý člen je právě zbytek po dělení i-tého členu $(i+1)$-ním členem. Tedy pro vhodné celé x_{i+1} (x_{i+1} použijeme místo q_{i+1} z předchozího výkladu EA):

$$a_{i+2} = a_i \pmod{a_{i+1}} = a_i - x_{i+1} \cdot a_{i+1}.$$

Euklidův algoritmus nám postupným dosazováním předchozích dvou členů posloupnosti za $i+2$-hý člen zároveň umožňuje vyjádřit předposlední člen jako celočíselnou lineární kombinaci prvních dvou členů. Tak například
$a_3 = a_1 - x_2 \cdot a_2 = a_1 - x_2 \cdot (a_0 - x_1 \cdot a_1) = (x_1 + 1) \cdot a_1 - x_2 \cdot a_0$
$a_4 = a_2 - x_3 \cdot a_3 = a_0 - x_1 \cdot a_1 - x_3 \cdot ((x_1+1) \cdot a_1 - x_2 \cdot a_0) = (x_2 \cdot x_3 + 1) \cdot a_1 - (x_1 + x_1 \cdot x_3) \cdot a_0$
atd. Toto pozorování můžeme vyjádřit tvrzením: Pro každá dvě různá kladná celá čísla a, b (omezujeme se na kladná celá čísla, protože $\mathrm{NSD}(a, b) = \mathrm{NSD}(|a|, |b|)$) existují celá čísla x, y tak, že

$$\mathrm{NSD}(a, b) = x \cdot a + y \cdot b.$$

Příklad 4.2. V příkladu 4.1 jsme počítali $\mathrm{NSD}(98, 56)$. Ukážeme, že každé číslo a_{i+1} je celočíselnou lineární kombinaci hodnot a_0 a a_1:
$a_0 = 98$
$a_1 = 56$
$a_2 = 42 = 98 - 1 \cdot 56$
$a_3 = 14 = 56 - 1 \cdot 42 - 56 - 1 \cdot (98 - 56) = 2 \cdot 56 - 1 \cdot 98, x = 2, y = -1$
▲

Pokud $n = b$ je prvočíslo, $0 < a < n$, pak zřejmě $\mathrm{NSD}(a, n) = 1$. Tedy podle předchozí úvahy existuje celé číslo x tak, že $1 = \mathrm{NSD}(a, n) = (x \cdot a) \pmod{n}$. Toto x můžeme volit z intervalu $0, 1, \ldots, n-1$. Multiplikativní inverzní prvek k nenulovému číslu ze Z_n jsme zavedli v definici 1.23. Pokud platí, že $\mathrm{NSD}(a, n) = 1$, můžeme Euklidův algoritmus použít pro nalezení tohoto inverzního prvku - tato varianta se nazývá *rozšířený Euklidův algoritmus* (Extended Euclidean algorithm, ozn. EEA).

Příklad 4.3. Pomocí Euklidova algoritmu určete hodnotu 18^{-1} v Z_{25}. Nejprve určíme NSD$(25, 18)$ proto, aby nám byl zřejmější postup hledání inverzního prvku k 18.

a_0	$a_0 = 25$
a_1	$a_1 = 18$
$a_2 = a_0 - q_1 \cdot a_1$	$a_2 = 25 - 1 \cdot 18 = 7$
$a_3 = a_1 - q_2 \cdot a_2$	$a_3 = 18 - 2 \cdot 7 = 4$
$a_4 = a_2 - q_3 \cdot a_3$	$a_4 = 7 - 1 \cdot 4 = 3$
$a_5 = a_3 - q_4 \cdot a_4$	$a_5 = 4 - 1 \cdot 3 = \mathbf{1} = \text{NSD}(25, 18)$

Protože je NSD$(25, 18) = 1$, budeme hledat 18^{-1} pomocí EEA:
$a_0 = 25$
$a_1 = 18$
$a_2 = 7 = 25 - 18$
$a_3 = 4 = 18 - 2 \cdot 7 = 18 - 2 \cdot (25 - 18) = 3 \cdot 18 - 2 \cdot 25$
$a_4 = 3 = 7 - 4 = 25 - 18 - (3 \cdot 18 - 2 \cdot 25) = 3 \cdot 25 - 4 \cdot 18$
$a_5 = \text{NSD}(25, 18) = 1 = 4 - 3 = 3 \cdot 18 - 2 \cdot 25 - (3 \cdot 25 - 4 \cdot 18) = 7 \cdot 18 - 5 \cdot 25$,
$x = 7, y = -5$.
Multiplikativní inverzní prvek k 18 je 7 $\pmod{25}$. Musí platit $a \cdot a^{-1} \equiv 1 \pmod{n}$ pro $a \in \mathbb{Z}, a \neq 0$, a to platí, $18 \cdot 7 \equiv 1 \pmod{25}$.
▲

Uveďme nyní nejčastější formulaci rozšířeného Euklidova algoritmu, který má složitost $O((log_2\ n)^2)$ bitových operací.
Vstup: dvě kladná celá čísla a, b kde $a > b$.
Výstup: $d = \text{NSD}(a, b)$ a c. č. x, y vyhovující $ax + by = d$.

```
if (b = 0)
{
    d = a; x = 1; y = 0;
    return (d, x, y);
}
x2 = 1; x1 = 0; y2 = 0; y1 = 1;
while(b > 0)
{
    q = floor(a/b); r = a - q * b; x = x2 - q * x1; y = y2 - q * y1;
    a = b; b = r; x2 = x1; x1 = x; y2 = y1; y1 = y;
}
d = a; x = x2; y = y2;
return (d, x, y).
```

Funkce $floor(x)$ představuje dolní celou část nějakého čísla x. Pokud $d > 1$, potom a^{-1} neexistuje, jinak je to x.

Příklady k procvičení

1. Pomocí Euklidova algoritmu nalezněte (postupujte podobně jako v příkladu 4.1):

- NSD$(24140, 16762)$,
- NSD$(4655, 12075)$,
- NSD$((x^3 + x + 1), (x^2 + x + 1))$ v $GF(2)$,
- NSD$((x^3 - x + 1), (x^2 + 1))$ v $GF(3)$.

2. Pomocí rozšířeného Euklidova algoritmu nalezněte multiplikativní inverzní prvek (pokud existuje):
   - $299 \pmod{323}$,
   - $1234 \pmod{4321}$,
   - $24140 \pmod{40902}$,
   - $550 \pmod{1769}$,
   - k polynomu $(x^3 + x + 1)$ v $GF(2^4)$ s $m(x) = (x^4 + x + 1)$.

4.3 Rychlé modulární umocňování

Dalším algoritmem, který si zde vysvětlíme, je *algoritmus pro rychlé modulární umocňování* nazývaný Repeated Square and Multiply Algorithm (ozn. RSMA). V poznámce 1.27 jsme se zmínili o této vlastnosti modulárního násobení:

$$[(a \pmod n) \cdot (b \pmod n)] \pmod n = (a \cdot b) \pmod n$$

a ukázali jsme si, jak lze této vlastnosti využít pro modulární umocňování. Např. $88^7 \pmod{187}$ spočítáme efektivněji jako

$$[(88^4 \pmod{187}) \cdot (88^2 \pmod{187}) \cdot (88^1 \pmod{187})] \pmod{187}.$$

To znamená, že můžeme redukovat modulo n jednotlivé mezivýsledky.

Pro mnoho kryptografických algoritmů (např. algoritmus RSA) a protokolů je algoritmus RSMA jeden ze zásadních algoritmů. Jeho bitová složitost je $O((log_2\ n)^2)$. Jedna verze RSMA je založena na faktu, že binární reprezentace čísla k je $\sum_{i=0}^{t} k_i 2^i$, kde $k_i \in \{0, 1\}$. Pak

$$\prod_{i=0}^{t} a_i^{k_i 2^i} = (a^{2^0})^{k_0} \cdot (a^{2^1})^{k_1} \cdot \ldots \cdot (a^{2^t})^{k_t}$$

Vstup: $a \in \mathbb{Z}_n$, c.č. $0 \leqq k < n$ jehož binární reprezentace je $k = \sum_{i=0}^{t} k_i 2^i$.
Výstup: $a^k \pmod n$.

```
b = 1;
if(k == 0)
    return (b);
A = a;
if(k_0 == 1)
    b = a;
for(i = 1; i < t; i++) // od LSB do MSB
{
    A = (A * A) mod n;
    if(k_i == 1)
        b = A * b mod n;
}
return (b);
```

Příklad 4.4. Vypočítejme $16243^{77} \pmod{622}$ pomocí RSMA.
Bitová reprezentace čísla $77 = 1 \cdot 2^6 + 0 \cdot 2^5 + 0 \cdot 2^4 + 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 64 + 8 + 4 + 1 = 1001101$. Pro účely algoritmu potřebujeme reprezentaci od nejméně významného bitu (tedy od LSB (nebo little-endian)), takže pak $77 = 1011001$.

i	0	1	2	3	4	5	6
k_i	1	0	1	1	0	0	1
A	16243	65	493	469	395	525	79
b	16243	16243	171	583	583	583	$\mathbf{29} = 16243^{77} \pmod{622}$

▲

Příklady k procvičení

1. Pomocí RSMA řešte:
   - $3^{33} \pmod{17}$,
   - $5^{6} \pmod{13}$,
   - $5^{35} \pmod{17}$,
   - $7^{6} \pmod{19}$,
   - $7^{11} \pmod{17}$,
   - $7^{19} \pmod{23}$.
2. Pomocí RSMA modulo ireducibilní polynom $x^5 + x^2 + 1$ řešte:
   - $(x+1)^{13}$,
   - $(x^2+1)^{11}$,
   - $(x^3+1)^{6}$.

Kapitola 5

Testování prvočíselnosti

Prvočísla jsou důležitou prerekvizitou pro mnohé kryptosystémy, zejména pro asymetrické, založené na použití dvojice klíčů - klíče veřejného a soukromého. Jako příklad si uveďme prvočísla p a q používaná v algoritmu RSA pro konstrukci modulu $n = p \cdot q$ nebo prvočíslo p použité v algoritmu pro dohodu na klíči Diffie-Hellman pro definici $\mathbb{Z}_p$.

Jednoduchou myšlenkou, ale výpočetně obtížnou, zda lze dané přirozené číslo vyjádřit jako součin dvou jiných přirozených čísel (zkoumáním prvočíselnosti), se matematici zabývali ještě v době, kdy neměli k dispozici nástroje, které by jim výpočty usnadnily. Nejznámějším z prvních algoritmů na zjištění všech prvočísel až po danou mez n je *Eratosthenovo síto*. Jeho principem je eliminace všech násobků prvočísel, která jsou menší než $\sqrt{n}$. Po eliminaci násobků prvočísel nám zbudou pouze prvočísla. Algoritmus má složitost $\mathrm{O}(n(\log\ n)(\log\ n))$.

Uveďme pár základních faktů o distribuci prvočísel. Známým faktem je, že prvočísel je nekonečně mnoho. Další fakta uvádíme formou vět bez důkazů, neboť tyto přesahují rámec skript.

Věta 5.1 (Prvočíselná). *Nechť $\pi(x)$ je počet prvočísel $\leqq x$. Potom*

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

To znamená, že pro velká x lze $\pi(x)$ aproximovat výrazem $x/\ln x$, kde $ln\ x$ je přirozený logaritmus. Například pro $x = 10^{10}, \pi(x) = 455052511$ a $\lfloor x/\ln x \rfloor = 434294281$.

Rozložení prvočísel je poměrně rovnoměrné, jak ilustrují následující tři výsledky.

Věta 5.2 (Dirichletova). *Jestliže $NSD(a, n) = 1$, pak existuje nekonečně mnoho prvočísel kongruentních $a \pmod{n}$.*

Lemma 5.3. *Nechť $\pi(x, n, a)$ je počet prvočísel $\leqq x$, která jsou kongruentní $a \pmod n$ a kde $NSD(a, n) = 1$. Potom*

$$\pi(x, n, a) \sim \frac{x}{\phi(n) ln\ x}.$$

Jinými slovy, prvočísla jsou zhruba rovnoměrně rozdělena mezi $\phi(n)$ zbytkových tříd v $\mathbb{Z}_n^*$, pro jakékoliv n.

Lemma 5.4. *Nechť p_n označuje n-té prvočíslo. Pak $p_n \sim n\ ln\ n$. Přesněji*

$$n\ ln\ n < p_n < n(ln\ n + ln\ ln\ n) \text{ pro } n \geqq 6.$$

Pro zajímavost uvádíme, že největší známé prvočíslo $2^{43112609} - 1$ má 12978189 desítkových číslic a bylo objeveno 08/2008. Jedná se o 45. známé Mersennovo číslo (Mersennova prvočísla mají tvar $2^p - 1$, kde p je prvočíslo), viz kapitola 5.3.

5.1 Nejjednodušší algoritmy

Nejpřirozenějším způsobem generování velkého prvočísla je vygenerovat náhodné kladné liché celé číslo n potřebné velikosti a otestovat, zda je to prvočíslo. Takovýto test může být sice velmi jednoduchý, jak ale uvidíme dále, pro velká celá čísla bude prakticky nepoužitelný.

Algoritmus *Trial division*, využívající zkušební dělení, je neomylný test prvočíselnosti, avšak efektivní pouze pro relativně malé hodnoty testovaných čísel (max. 24 číslic). Základní verze algoritmu spočívá v postupném vydělení testovaného čísla n všemi čísly m takovými, že $1 < m < n$. Pokud je pro některé m výsledek dělení beze zbytku, číslo n není prvočíslo.

V základní podobě je algoritmus velmi neefektivní, některá dělení jsou zbytečná. Vylepšení lze dosáhnout tak, že dělíme pouze čísly $m \leqq \sqrt{n}$ a vynecháme sudá čísla. Pokud lze n vyjádřit jako součin dvou (nebo více) přirozených čísel, pak jedno z nich musí být menší nebo rovno jeho odmocnině. V ideálním případě, pokud máme seznam prvočísel v rozsahu $\langle 1, \sqrt{n} \rangle$, eliminujeme počet operací na minimum, dělíme pouze prvočísly z tohoto seznamu. Vyžaduje $\pi(\sqrt{(n)})$ pokusných dělení, řádová složitost je $(O(2^b)$, kde b je počet bitů c.č. n.

Příklad 5.5. Je dáno číslo 211. Postupně ho musíme dělit prvočísly $2, 3, 5, 7, 11, 13$, protože $\sqrt{211} = 14.5258$. Jak vidíme, číslo 211 je prvočíslo, protože:
$211 \pmod 2 = 1$,
$211 \pmod 3 = 1$,
$211 \pmod 5 = 1$,
$211 \pmod 7 = 1$,
$211 \pmod{11} = 2$,
$211 \pmod{13} = 3$.
▲

Variantou algoritmu Trial division je algoritmus *Wheel factorization*. Abychom eliminovali nutnost znát prvočísla až do $\sqrt{n}$, dělíme testované číslo pouze několika prvními k prvočísly, např. $p_1, p_2, \ldots, p_k$ a pak dané číslo dělíme čísly nesoudělnými s těmito k prvočísly až do $\sqrt{n}$. Čísla, kterými se dělí testované číslo, jsou s velkou pravděpodobností prvočísla, ale ne nutně.

Platí, že více než třetina složených čísel je dělitelná 3 a více než pětina jich je dělitelná 5. Pokud bychom např. nechtěli dělit násobky $2, 3, 5$, a 7, můžeme ušetřit si více než 77% práce. Obecně existuje v intervalu $\langle 2, n\rangle$ $n - \pi(n) - 1$ složených čísel a okolo $n\Pi_{k=1}^{i}(1 - \frac{1}{p_k}) - \pi(n) - 1$ složených čísel je nesoudělných s prvními i prvočísly [6]. Složitost algoritmu je srovnatelná se složitostí algoritmu Trial division (použijeme-li dělení prvočísly až do $\sqrt{n}$).

Příklad 5.6. Je zadáno číslo 3331, $\sqrt{3331} \doteq 57.7$. Budeme ho dělit např. prvočísly $2, 3$, a pak čísly, která jsou kongruentní 1 a 5 (mod 6) protože $2 \cdot 3 = 6$. Musíme odstranit všechny násobky čísel $2, 3$.

1	**2**	**3**	4	**5**	6
7	8	9	10	**11**	12
13	14	15	16	**17**	18
19	20	21	22	**23**	24
25	26	27	28	**29**	30
31	32	33	34	**35**	36
37	38	39	40	**41**	42
43	45	46	47	**49**	50
51	52	53	54	**55**	56
57	58	59	60	**61**	62

Vidíme, že číslo budeme dělit čísly $2, 3, 5, 7, 11, 13, 17, 19, 23, 25, 29, 31, 35, 37, 41, 43, 49, 51, 55, 57$, mezi kterými se vyskytují také čísla složená. Zjistíme, že číslo 3331 je prvočíslo.

▲

Všimněte si, že jednoduchým použitím prvočísel 2 a 3 byly odstraněny 2/3 složených čísel. Použijeme-li větší počet prvočísel, např. $2, 3, 5, 7$, odstraníme přes 77% složených čísel. Pokud bychom se domnívali, že využití většího počtu prvočísel algoritmus zefektivní, tedy odstraníme větší počet složených čísel, kterými bychom dělili testované číslo, není tomu tak. Např. chceme-li odstranit 90% složených čísel, musíme použít prvočísla až po 251.

5.2 Pravděpodobnostní testy

Pojďme se věnovat efektivnějším způsobům testování prvočíselnosti. Obecný postup bude následující:

1. Vygeneruj jako kandidáta na prvočíslo liché n potřebné velikosti.

2. Otestuj, zda je n prvočíslo.

3. Když n je složené číslo, přejdi k bodu 1.

Ve druhém kroku tohoto základního algoritmu můžeme použít test, který prokazatelně určí kandidáta jako prvočíslo (v angličtině se pro takovéto prvočíslo používá pojem „provably prime“). Tyto testy jsou ale výpočetně obtížnější než testy, které určí kandidáta jako prvočíslo s jistou pravděpodobností (v angličtině se pro takovéto prvočíslo používá pojem „probably prime“).

Testy označované jako testy složenosti (compositeness tests), nebo také pravděpdobnostní testy, určí kandidáta, který není prvočíslem, spolehlivě jako číslo složené. Neposkytují však matematický důkaz, že kandidát označený jako probably prime, je skutečně prvočíslo. Tyto testy se provádějí opakovaně. Volíme parametr t, pokud tyto testy proběhnou t-krát pro složené číslo n, pravděpodobnost, že n bude deklarováno jako prvočíslo ve všech t pokusech, je nejvýše $(1/2)^t$.

5.2.1 Fermatův test

Jako první uveďme *Fermatův test prvočíselnosti*. Tento test není typickým pravděpodobnostním testem, protože nedokáže rozlišit prvočísla od speciálních složených čísel nazvaných Carmichaelova čísla, bývá spíše označován jako test složenosti. Nicméně je to test důležitý. Vychází z tzv. malé Fermatovy věty, jejíž význam pro testy prvočíselnosti (resp. lépe složenosti) vyplývá z toho, že modulární umocňování umíme provádět efektivně (viz kapitoly 4.3 a 6).

Malá Fermatova věta, viz kap 6.1, říká, že jestliže n je prvočíslo, a je kladné celé číslo nesoudělné s n, $1 \leqq a \leqq n-1$, potom $a^{n-1} \equiv 1 \pmod{n}$. Pokud testujeme složenost čísla n, pak nalezení jakéhokoliv a, pro které uvedená kongruence neplatí, stačí jako důkaz složenosti čísla n. Pokud ale tato kongruence platí, pak ještě není řečeno, že n je prvočíslo. Celý test provedeme znovu s jiným a, resp. test provádíme t-krát pro t různých čísel a. Řádová složitost algoritmu se uvádí jako $O(t \cdot log\ n)$. Zaveďme následující pojmy.

Definice 5.7. Nechť je n liché složené číslo. Celé číslo a, $1 \leqq a \leqq n-1$ takové, že $a^{n-1} \not\equiv 1 \pmod{n}$, se nazývá *Fermatův svědek složenosti* (Fermat's witness) čísla n.

Definice 5.8. Nechť je n liché složené číslo a a je celé číslo $1 \leqq a \leqq n-1$. Potom n se nazývá *pseudoprvočíslo* (pseudoprime) vzhledem k bázi a, jestliže $a^{n-1} \equiv 1 \pmod{n}$. Číslo a se nazývá *Fermatův lhář* (Fertmat's liar) vzhledem k n.

Příklad 5.9. Např. složené číslo $n = 341 (341 = 11 \cdot 31)$ je pseudoprvočíslo vzhledem k bázi $a = 2$, protože $2^{340} \equiv 1 \pmod{341}$.
▲

Fermatův test prvočíselnosti tedy formulujme takto:
Vstup: liché c.č. $n \geqq 3$ a parametr $t \geqq 1$.
Výstup: odpověď na otázku „je n prvočíslo?“

```
for(i = 1; i <= t; i++)
{
    vyber náhodné int a;
    r = a^(n-1) mod n; // (pomocí RSMA)
    if(r != 1) then
        return ("složené");
        break;
}
return ("prvočíslo");
```

Složených čísel, která jsou pseudoprvočísly vzhledem k libovolné bázi a, kde $\mathrm{NSD}(a, n) = 1$, je nekonečně mnoho. Tato čísla se nazývají Carmichaelova čísla. Carmichaelova čísla jsou poměrně řídká (pouze 2163 jich je menších než 25 000 000 000, nebo 105212 jich je menších než 10^{15}). Nejmenší Carmichaelovo číslo je $561 = 3 \cdot 11 \cdot 17$, dalšími jsou $1105, 1729, 2465, 2821, \ldots$.

Definice 5.10. *Carmichaelovo číslo* n je složené číslo takové, že $a^{n-1} \equiv 1 \pmod{n}$ pro všechna a, pro která $\mathrm{NSD}(a, n) = 1$.

Teoreticky můžeme pro testování prvočíselnosti velkého náhodného lichého c. č. použít v praxi i Fermatův test, byť se to běžně nedělá. Pokud generujeme náhodné liché celé číslo, je pravděpodobnost, že to bude pseudoprvočíslo (složené kladné celé číslo n, pro které je alespoň pro jednu bázi a určeno n chybně jako prvočíslo) tím menší, čím je větší počet desítkových číslic daného celého čísla. Tuto informaci můžeme přehledně dokumentovat tabulkou 5.1, kde $P(x)$ vyjadřuje pravděpodobnost, že složené liché c.č. n, $1 < n \leqq x$, bude pseudoprvočíslo vzhledem ke každé bázi a (pro detaily viz [4]).

Počet cifer x	Pravděpodobnost $P(x)$ že n je pseudoprvočíslo
60	0.0716
100	0.0000000277
120	$5.28 * 10^{-12}$
200	$3.85 * 10^{-27}$
500	$2.3 * 10^{-55}$
1000	$1.2 * 10^{-123}$

Tab. 5.1 Pravděpodobnost, že c.č. n je pseudoprvočíslo

5.2.2 Miller-Rabinův test

Pravděpodobnostním testem, který se v praxi používá velmi často, je *Miller-Rabinův test*. Tento test využívá faktu, že je-li c.č. n prvočíslo, pak má právě dvě odmocniny jedničky modulo n, a to 1 a -1 . Pojem odmocnina modulo n a další nezbytné teoretické předpoklady Miller-Rabinova testu si nyní postupně vysvětlíme.

Definice 5.11. Nechť $a \in \mathbb{Z}_n^*$. Číslo a se nazývá *kvadratický zbytek* (quadratic residue) modulo n jestliže existuje $x \in \mathbb{Z}_n^*$ takové že $x^2 \equiv a \pmod{n}$. Množina kvadratických zbytků modulo n bude označována Q_n.

Příklad 5.12. Např. $Q_7 = \{1, 2, 4\}$, protože $\mathbb{Z}_7^* = \{1, 2, 3, 4, 5, 6\}$ a dále:
$1^2 \pmod 7 = 1$,
$2^2 \pmod 7 = 4$,
$3^2 \pmod 7 = 2$,
$4^2 \pmod 7 = 2$,
$5^2 \pmod 7 = 4$,
$6^2 \pmod 7 = 1$.
▲

Platí: Nechť p je liché prvočíslo a g je generátor cyklické grupy $\mathbb{Z}_p^*$. Potom a je kvadratický zbytek modulo p právě tehdy, když $a = g^i \pmod p$, kde i je sudé celé číslo. Platí, že $|Q_p| = (p-1)/2$.

Příklad 5.13. Nechť $g = 6$ je generátor $\mathbb{Z}_{13}^*$. Mocniny g jsou uvedeny v následující tabulce. Pak $Q_{13} = \{1, 3, 4, 9, 10, 12\}$ (pro sudé i).

i	0	1	2	3	4	5	6	7	8	9	10	11
$a = g^i \pmod{13}$	1	6	10	8	9	2	12	7	3	5	4	11

Ověříme, zda je množina $Q_{13} = \{1, 3, 4, 9, 10, 12\}$ určena správně v $\mathbb{Z}_{13}^* = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12\}$:

$1^2 \pmod{13} = 1$
$2^2 \pmod{13} = 4$
$3^2 \pmod{13} = 9$
$4^2 \pmod{13} = 3$
$5^2 \pmod{13} = 12$
$6^2 \pmod{13} = 10$
$7^2 \pmod{13} = 10$
$8^2 \pmod{13} = 12$
$9^2 \pmod{13} = 3$
$10^2 \pmod{13} = 9$
$11^2 \pmod{13} = 4$
$12^2 \pmod{13} = 1$

▲

Definice 5.14. Nechť $a \in Q_n$. Jestliže $x \in \mathbb{Z}_n^*$ splňuje $x^2 \equiv a \pmod{n}$ pak se x nazývá (druhá) *odmocnina* (square root) čísla a modulo n.

Platí:

i) Jestliže p je liché prvočíslo, pak $a \in Q_p$ má přesně dvě druhé odmocniny modulo p.

ii) Obecněji, nechť $n = {p_1}^{e_1} \cdot {p_2}^{e_2} \cdot \ldots \cdot {p_k}^{e_k}$, kde $p_1, p_2, \ldots, p_k$ jsou navzájem různá prvočísla a $e_i \geqq 1$. Pokud $a \in Q_n$, pak a má přesně 2^k různých odmocnin modulo n.

Příklad 5.15. Odmocniny č. 4 (mod 13) jsou č. 2 a 11, viz předchozí příklad.
Odmocniny č. 12 (mod 37) jsou č. 7 a 30.
Odmocniny 121 (mod 315) jsou $11, 74, 101, 151, 164, 214, 241$, a 304.
▲

Zesílení Malé Fermatovy věty nám poskytuje možnost výhodnějšího testování prvočíselnosti (složenosti):

Lemma 5.16. *Pro libovolné liché prvočíslo p a libovolné celé číslo a, $1 \leqq a \leqq p-1$, $NSD(a,p) = 1$ platí:*

$$a^{\frac{p-1}{2}} \equiv \pm 1 \pmod{p}.$$

Důkaz. Z Fermatovy věty platí $a^{p-1} - 1 \equiv 0 \pmod{p}$. Dále

$$(a^{p-1} - 1) = (a^{\frac{p-1}{2}} - 1) \cdot (a^{\frac{p-1}{2}} + 1).$$

Protože je p prvočíslo, musí dělit i některého z uvedených činitelů. □

Příklad 5.17. Test založený na výpočtu $a^{\frac{n-1}{2}} \pmod{n}$ a kontrole, zda je výsledek 1 nebo -1, vyloučí první Carmichaelovo číslo $561 = 3 \cdot 11 \cdot 17$. Pro např. $a = 5$ dostaneme $5^{\frac{561-1}{2}} = 5^{280} \equiv 67 \pmod{561}$.

Avšak tento test neodhalí například třetí Carmichaelovo číslo $n = 1729 = 7 \cdot 13 \cdot 19$, protože např. opět pro $a = 5$ dostaneme $5^{\frac{1729-1}{2}} = 5^{864} \equiv 1 \pmod{1729}$.
▲

Proto je třeba podmínku, kterou budeme testovat, ještě více zesílit. Pro kandidáta na prvočíslo, tedy pro liché celé číslo $n \geqq 3$, mějme sudé $n-1$, které můžeme vyjádřit jako součin určité mocniny čísla 2 a lichého čísla r, tj $n - 1 = 2^s r, s > 0$. Algoritmus Miller-Rabin pak obsahuje výpočet zbytků po dělení číslem n této posloupnosti mocnin $a^r, a^{2r}, \ldots, a^{2^{s-1}r}$ pro $1 \leqq a \leqq n-1$.

Jestliže je n prvočíslo, podle Fermatovy věty víme, že $a^{n-1} \pmod{n} = a^{2^s r} \pmod{n} = 1$. Mohou tak existovat dřívější prvky z posloupnosti $a^r, a^{2r}, \ldots, a^{2^{s-1}r}$, které dají po dělení n zbytek 1. Algoritmus navržený Millerem a Rabinem tedy využívá následujícího tvrzení (připomínáme, že $-a = n - a$ pro každé $a \in Z_n$, viz kapitola 3.2):

Lemma 5.18. *Nechť p je liché prvočíslo a nechť $p-1 = 2^s r$, kde r je liché. Nechť a je celé č. takové že $NSD(a,p) = 1$. Potom buď $a^r \equiv 1 \pmod{p}$ nebo $a^{2^j r} \equiv -1 \pmod{p}$ pro nějaké $j, 0 \leqq j \leqq s-1$.*

Důkaz. Z Fermatovy věty platí $a^{p-1} - 1 \equiv 0 \pmod{p}$. Dále

$$(a^{p-1} - 1) = (a^r - 1) \cdot \Pi_{j=0}^{s-1}(a^{2^j \cdot r} + 1).$$

Protože je p prvočíslo, musí p dělit i některého z uvedených činitelů. □

Definice 5.19. Nechť n je liché složené číslo a nechť $n-1 = 2^s r$, kde r je liché. Nechť a je celé č. $1 \leqq a \leqq n-1$.

- Jestliže $a^r \not\equiv 1 \pmod{n}$ a jestliže $a^{2^j r} \not\equiv -1 \pmod{n}$ pro každé $j, 0 \leqq j \leqq \leqq s-1$, pak se a nazývá *silný svědek složenosti* (strong witness) čísla n.
- Jinak jestliže $a^r \equiv 1 \pmod{n}$ nebo $a^{2^j r} \equiv -1 \pmod{n}$ pro nějaké $j, 0 \leqq j \leqq \leqq s-1$, pak se n nazývá *silné pseudoprvočíslo* vzhledem k bázi a. Celé číslo a se nazývá *silný lhář* (strong liar) vzhledem k n.

Příklad 5.20. Mějme $n = 7 \cdot 13$. Pak $91 - 1 = 90 = 2 \cdot 45, s = 1$ a $r = 45$. Protože $9^r = 9^{45} \equiv 1 \pmod{91}$, je číslo 91 silné psedudoprvočíslo vzhledem k bázi 9. Množina všech silných lhářů pro 91 je

$$\{1, 9, 10, 12, 16, 17, 22, 29, 38, 53, 62, 69, 74, 75, 79, 81, 82, 90\}.$$

▲

Lemma 5.18 může být použito jako základ pro pravděpodobnostní testování díky následujícímu tvrzení. Jestliže je n liché složené číslo, potom nejvýše $\frac{1}{4}$ ze všech bází $1 \leqq a \leqq n-1$ je silnými lháři pro n. Jestliže $n \neq 9$, počet silných lhářů pro n je nejvýše $\frac{\phi(n)}{4}$.

Pro jakékoliv liché složené číslo n je pravděpodobnost, že bude určeno Miller-Rabinovým testem jako prvočíslo, menší než $(1/4)^t$. Např. je-li $t = 10$, pak pravděpodobnost, že složené n je určeno jako prvočíslo je 0.00000095367431640625. Řádová složitost algoritmu se uvádí jako $O(t \cdot log^3\ n)$.

Formulace Miller-Rabinova testu:
Vstup: liché c.č. $n \geqq 3$ a parametr $t \geqq 1$.
Výstup: odpověď na otázku „je n prvočíslo?“

```
vyjádři n ve tvaru n-1 = 2^s r, r liché;
for(i = 1; i <= t; i++)
{
    vyber náhodné int a;
    y = a^r mod n;
    if(y == 1)
        return ("prvočíslo");
   for(j = 0; j < s; j++)
    {
      if (a^(2^j r) mod n == n - 1)   (*)
         return("prvočíslo");
    }
    return ("složené");
}
```

Příklad 5.21. Nechť $n = 29, n - 1 = 2^s \cdot r, r$ liché, pak $28 = 2^2 \cdot 7, s = 2, r = 7$. Nechť $t = 3$.

i=1, zvolíme $a = 10$,
vypočteme $10^7 \pmod{29} = 17$ (není ani 1 ani 28).
Pokračujeme $(10^7)^2 \pmod{29} = 28 \Rightarrow$ „prvočíslo“.

i=2, zvolíme $a = 3$,
vypočteme $3^7 \pmod{29} = 14$ (není ani 1 ani 28).
Pokračujeme $(3^7)^2 \pmod{29} = 28 \Rightarrow$ „prvočíslo“.

i=3, zvolíme $a = 5$,
vypočteme $5^7 \pmod{29} = 28 \Rightarrow$ „prvočíslo“.

Pokud provedeme test pro všechna $a < 29$, dostaneme vždy výsledek „prvočíslo“, tedy $n = 29$ je prvočíslo.
▲

Příklad 5.22. Nechť $n = 221 = 13 \cdot 17, n - 1 = 2^s \cdot r, r$ liché, pak $220 = 2^2 \cdot 55, s = = 2, r = 55$. Nechť $t = 3$.

i=1, zvolíme $a = 10$,
vypočteme $10^{55} \pmod{221} = 192$ (není ani 1 ani 220).
Pokračujeme $(10^{55})^2 \pmod{221} = 9$ (není ani 1 ani 220).
Protože jsme však vypočetli $a^{2^j r} \pmod{n}$ pro všechna $j < s$, test vrátí, že č. 221 je „složené“.

i=2, zvolíme $a = 5$,
vypočteme $5^{55} \pmod{221} = 112$ (není ani 1 ani 220).
Pokračujeme $(5^{55})^2 \pmod{221} = 168$ (není ani 1 ani 220).
Protože jsme však vypočetli $a^{2^j r} \pmod{n}$ pro všechna $j < s$, test vrátí, že č. 221 je „složené“.

i=3, zvolíme $a = 47$,
vypočteme $47^{55} \pmod{221} = 63$ (není ani 1 ani 220)
Pokračujeme $(47^{55})^2 \pmod{221} = 220 \Rightarrow$ „prvočíslo“.

Pro $n = 221$ je v intervalu $1 < a < n$ šest čísel a, pro která dostaneme výsledek „prvočíslo“, ačkoliv n je složené. Jsou to $a \in \{1, 21, 47, 174, 200, 220\}$.
▲

Na závěr uveďme, že se mezi pravděpodobnostní testy ještě obvykle řadí algoritmus Solovay-Strassen, který se však prakticky nepoužívá.

5.3 Testy dokazující prvočíselnost

V této části textu představíme alespoň jeden test ze skupiny testů, které jsou schopny dokázat prvočíselnost kladného celočíselného kandidáta na prvočíslo. Tyto testy jsou obecně výpočetně náročnější než testy z předchozí skupiny testů. Proto je vhodné před jejich vlastním použitím otestovat kandidáta na prvočíslo, tedy kladné celé číslo n, některým z pravděpodobnostních testů (např. Miller-Rabinovým testem).

Jako zástupce této skupiny testů si uvedeme *test Lucas-Lehmerův*, který je efektivním testem prvočíselnosti pro speciální čísla, takzvaná Mersennova čísla (o Mersennově prvočíslu jsme se zmínili v úvodu celé kapitoly 5).

Definice 5.23. Nechť $s \geqq 2$ je celé číslo. Mersennovo číslo je číslo ve tvaru $2^s - 1$. Jestliže je $2^s - 1$ prvočíslo, pak se nazývá Mersennovo provočíslo.

Lemma 5.24. *Nechť $s \geqq 3$. Mersennovo číslo $n = 2^s - 1$ je prvočíslem právě tehdy, když:*

i) s je prvočíslo,

ii) a posloupnost celých čísel definovaná: $u_0 = 4$ a $u_{k+1} = (u_k^2 - 2) \pmod{n}$ pro $k \geqq 0$ splňuje $u_{s-2} = 0$.

Prvních deset Mersennových prvočísel bylo určeno do konce 19. století, viz tabulka 5.2:

#	1	2	3	4	5	6	7	8	9	10	11	12
s	2	3	5	7	13	17	19	31	61	89	107	127
počet cifer	1	1	2	3	4	6	6	10	19	27	33	39
rok	–	–	–	–	1456	1588	1588	1772	1883	1911	1914	1876

Tab. 5.2 Prvních 12 Mersennových prvočísel

Lucas-Lehmerův test prvočíselnosti formulujme takto:
Vstup: Mersennovo číslo $n = 2^s - 1$, $s \geqq 3$.
Výstup: odpověď na otázku „je n prvočíslo?“

```
1. Použij Trial division pro kontrolu, zda s má dělitele z intervalu
   ⟨2, ⌊√2⌋⟩.
   Pokud ano, return ("složené");
2. u = 4;
3. for(k = 1; k < s-2; k++)
   {
       u = (u * u - 2) mod n;
   }
4. if(u == 0)
      return ("prvočíslo");
   else
      return ("složené");
```

Příklad 5.25. Otestujme, zda je Mersennovo číslo $31 = 2^5 - 1$ prvočíslo.

k	1	2	3		
u	14	8	0	$\rightarrow 31$	je prvočíslo

▲

Dalším testem dokazujícím prvočíselnost je Pepinův test, který je založen na tzv. Pocklingtonově větě. Zájemce odkazujeme na [5, 9]. Dalším algoritmem, který nesmíme opomenout alespoň zmínit, je algoritmus z roku 2002 označovaný jako AKS, podle svých autorů (Agrawal, Kayal a Saxena) [1]. Jedná se o polynomiální deterministický algoritmus, který se neopírá o žádné neprokázané předpoklady, ani nevyžaduje faktorizaci. AKS používá další zobecnění Malé Fermatovy věty: Jestliže n je prvočíslo, potom pro všechna $a, 1 \leqq a < n$ platí

$$(x - a)^n = (x^n - a)(\mathrm{mod}\ n, x^r - 1),$$

kde $p(x)(\mathrm{mod}\ n, x^r - 1)$ je polynom, který je zbytkem po dělení koeficientů modulo n a mocnin x modulo r.

Příklady k procvičení

1. Otestujte čísla $127, 341$ a 1105 všemi popsanými algoritmy.

2. Ukažte, že je číslo 2047 silné pseudoprvočíslo vzhledem k bázi 2. Použijte Miller-Rabinův algoritmus.

3. Nalezněte množinu Q_{11} pro $\mathbb{Z}_{11}^*$ a odmocninu libovolných dvou $a \in \mathbb{Z}_{11}^*$ modulo 11.

4. Nalezněte množinu Q_{25} pro $\mathbb{Z}_{25}^*$ a odmocninu libovolných dvou $a \in \mathbb{Z}_{25}^*$ modulo 25.

5. Nalezněte množinu Q_{21} pro $\mathbb{Z}_{21}^*$ a odmocninu libovolných dvou $a \in \mathbb{Z}_{21}^*$ modulo 21. Pozn.: ověřte, zda je grupa $\mathbb{Z}_{21}^*$ cyklická.

Kapitola 6

Důležité věty a problémy

Tato kapitola uvádí důležité věty a problémy, které jsou teoretickým základem mnoha algoritmů a jejich dílčích operací a ukazuje některé možnosti jejich využití.

6.1 Malá Fermatova věta

Velká Fermatova věta (Fermat's Big Theorem nebo také Fermat's Last Theorem) říká, že $a^n+b^n = c^n$ nemá řešení pro kladná celá čísla a, b, c pro $n > 2$. Byla dokázána Andrew Wilesem v roce 1995, teprve 350 let po jejím prvním zformulování Pierrem de Fermatem.

Tato slavná Fermatova věta však není předmětem našeho zájmu, důležitou je pro nás tzv. Malá Fermatova věta. Dosud jsme používali malou Fermatovu větu, resp. také její důsledky, aniž bychom uvedli její důkaz, což nyní učiníme.

Věta 6.1 (Malá Fermatova věta). *Pro každé prvočíslo p a každé kladné celé číslo a nesoudělné s p platí*

$$a^{p-1} \equiv 1 \pmod{p}.$$

Důkaz. Víme, že $\mathbb{Z}_p = \{0, 1, \ldots, p-1\}$, a že pokud jsou čísla z množiny $\mathbb{Z}_p$ vynásobena a modulo p, je výsledkem množina prvků z $\mathbb{Z}_p$ v nějakém pořadí. Tedy:

a) Platí $a \cdot 0 \equiv 0 \pmod{p}$.

b) Dále platí že $p-1$ čísel $\{a \pmod{p}, 2a \pmod{p}, \ldots, (p-1)a \pmod{p}\}$ jsou čísla z množiny Z_p v nějakém pořadí. Vynásobíme spolu tato čísla:

$$\begin{gathered} a \cdot 2a \cdot \ldots \cdot ((p-1)a) \equiv \\ \equiv [(a \pmod{p}) \cdot (2a \pmod{p}) \cdot \ldots \cdot ((p-1)a \pmod{p})] \pmod{p} \equiv \\ \equiv [1 \cdot 2 \cdot \ldots \cdot (p-1)] \pmod{p} \equiv (p-1)! \pmod{p}. \end{gathered}$$

Ale $a \cdot 2a \cdot \ldots \cdot ((p-1)a) = (p-1)!a^{p-1}$. Proto

$$(p-1)!a^{p-1} \equiv (p-1)! \pmod{p}.$$

Vykrátíme $(p-1)!$ a dostaneme $a^{p-1} \equiv 1 \pmod{p}$.

□

Příklad 6.2. $a = 7, p = 19$,
$7^2 = 49 \equiv 11 \pmod{19}$,
$7^4 = (7^2)^2 \equiv 11^2 \equiv 7 \pmod{19}$,
$7^8 = (7^4)^2 \equiv 7^2 \equiv 11 \pmod{19}$,
$7^{16} = (7^8)^2 \equiv 11^2 \equiv 7 \pmod{19}$,
$a^{p-1} = 7^{18} = 7^{16} \cdot 7^2 \equiv 7 \cdot 11 \equiv 1 \pmod{19}$.
▲

Poznámka 6.3. Z MFV plyne:

- Jiná formulace MFV: Je-li p prvočíslo, pak pro každé celé číslo a platí
$$a^p \equiv a \pmod{p}.$$
- Jestliže $r \equiv s \pmod{(p-1)}$ pak $a^r \equiv a^s \pmod{p}$ pro všechna a. Jinak řečeno pokud počítáme modulo p, exponenty mohou být redukovány modul $p-1$.

Příklad 6.4. $p = 5, a = 3, 3^5 = 243 \equiv 3 \pmod{5}$,
$p = 5, a = 10, 10^5 = 100000 \equiv 10 \pmod{5} \equiv 0 \pmod{5}$.
▲

Příklad 6.5. Z předchozí poznámky plyne, že MFV můžeme použít pro snazší výpočet modulárního umocňování $a^m \pmod{p}$ kde p je prvočíslo. Pro zjednodušení výpočtu můžeme použít následující:

(1) a nahradit $a \pmod{p}$,

(2) m nahradit $m \pmod{(p-1)}$.

Chceme vypočítat:
$$1234^{7865435} \pmod{11}.$$
Ad (1) platí, že $1234 \equiv 2 \pmod{11}$. Protože $\mathrm{NSD}(2, 11) = 1$ dostaneme z MFV $2^{10} \equiv 1 \pmod{11}$.
Nyní z (2) máme $7865435 \equiv 5 \pmod{10}$, tj. $7865435 = (786543) \cdot 10 + 5$, proto

$$\begin{aligned} 2^{7865435} &\equiv 2^{(786543)\cdot 10+5} \pmod{11} \\ &\equiv \left(2^{10}\right)^{786543} \cdot 2^5 \pmod{11} \\ &\equiv 1^{786543} \cdot 2^5 \pmod{11} \\ &\equiv 2^5 \pmod{11}, \end{aligned}$$

a $2^5 = 32 \equiv 10 \pmod{11}$. Proto
$$1234^{7865435} \equiv 10 \pmod{11}.$$
▲

6.2 Eulerova věta

V definici 1.15 jsme zavedli Eulerovu funkci a pak jsme si uvedli její vlastnosti. Připomínáme, že pokud p a q jsou dvě prvočísla a n je c.č., platí

$$n = p \cdot q, \phi(n) = \phi(p \cdot q) = \phi(p) \cdot \phi(q) = (p-1) \cdot (q-1).$$

Předpokládejme že množina zbytků v Z_n je $\{0, 1, \ldots, (p \cdot q - 1)\}$. Zbytky, které jsou soudělné s n jsou množiny $\{p, 2p, \ldots, (q-1)p\}$, $\{q, 2q, , (p-1)q\}$ a 0.
Tedy je-li $n = p \cdot q$, pak

$$\phi(n) = p \cdot q - [(q-1) + (p-1) + 1] =$$

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

Zobecněním Malé Fermatovy věty je Eulerova věta (známá jako Euler's theorem, Fermat-Euler theorem nebo také Euler's totient theorem) (jejím zobecněním je Carmichaelova věta, jejíž dopad na kryptografii ale není tak velký, abychom jí věnovali pozornost).

Eulerova věta nám opět poskytuje možnost redukovat velké mocniny modulo n, tedy provádět efektivně modulární umocňování. Nejprve si uvedeme definice, které jsou nezbytné k důkazu Eulerovy věty.

Definice 6.6. Nechť n je kladné celé číslo. Množina celých čísel $\{a_1, a_2, \ldots, a_n\}$ se nazývá *úplný systém zbytků modulo* n, jestliže obsahuje právě jeden prvek z každé zbytkové třídy modulo n.

Příklad 6.7. Nechť $n = 4$. Pak $\{-12, 9, -6, 15\}$ je úplný systém zbytků modulo 4, protože $-12 \in [0]_4, 9 \in [1]_4, -6 \in [2]_4, 15 \in [3]_4$.
▲

Definice 6.8. Nechť n je kladné celé číslo, pak $\phi(n)$ je počet zbytkových tříd modulo n, který je nesoudělný s n. Množina celých čísel $\{a_1, a_2, \ldots, a_{\phi(n)}\}$ se nazývá *redukovaný systém zbytků modulo* n, jestliže obsahuje právě jeden prvek z každé zbytkové třídy modulo n, který je nesoudělný s n.

Příklad 6.9. Říkáme, že zbytková třída $[a]_n$ je nesoudělná s n, jestliže je každý prvek z $[a]_n$ nesoudělný s n. Např5. zbytkové třídy $[1]_{10}, [3]_{10}, [7]_{10}, [9]_{10}$ jsou zbytkové třídy nesoudělné s 10. Pak $\{-29, -17, 17, 39\}$ je redukovaný systém zbytků modulo 10, protože $-29 \in [1]_{10}, -17 \in [3]_{10}, 17 \in [7]_{10}, 39 \in [9]_{10}$.

Nejjednodušším redukovaným systémem zbytků modulo n je množina celých čísel $\{0, \ldots, n-1\}$ nesoudělných s n.
▲

Věta 6.10 (Eulerova věta). *Pro každé celé číslo $n \geqq 2$ a každé celé číslo a, které je nesoudělné s n, platí*

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

Důkaz. Nechť $r_1, r_2, \ldots, r_{\phi(n)}$ je redukovaný systém zbytků modulo n. Pak $ar_1, ar_2, \ldots, ar_{\phi(n)}$ je také redukovaný systém zbytků modulo n. Potom

$$(ar_1)(ar_2)\ldots(ar_{\phi(n)}) \equiv r_1 r_2 \ldots r_{\phi(n)} \pmod{n},$$

protože je-li $ar_1, ar_2, \ldots, ar_{\phi(n)}$ redukovaný systém zbytků modulo n, musí být kongruentní $r_1, r_2, \ldots, r_{\phi(n)}$ (v nějakém pořadí). Proto

$$a^{\phi(n)} r_1 r_2 \ldots r_{\phi(n)} \equiv r_1 r_2 \ldots r_{\phi(n)} \pmod{n},$$

což implikuje

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

□

Poznámka 6.11. Z Eulerovy věty plyne: Jestliže n je součin dvou různých prvočísel a $r \equiv s \pmod{\phi(n)}$, pak $a^r \equiv a^s \pmod{n}$ pro všechna celá čísla a. Jinak řečeno pokud počítáme modulo n, exponenty mohou být redukovány modul $\phi(n)$.

Příklad 6.12. $N = 10, a = 3, \phi(N) = 4, 3^4 = 81 \equiv 1 \pmod{10}$,
$N = 11, a = 2, \phi(N) = 10, 2^{10} = 1024 \equiv 1 \pmod{11}$.
▲

Příklad 6.13. Ukážeme si, jak můžeme Eulerovu větu využít pro snazší výpočet modulárního umocňování $a^m \pmod{n}$. Pro zjednodušení výpočtu můžeme použít následující:

(1) a nahradit $a \pmod{n}$,

(2) m nahradit $m \pmod{\phi(\cancel{m})}$.

Chceme vypočítat $1234^{5678} \pmod{11}$. Vidíme, že $\mathrm{NSD}(1234, 11) = 1$. Ad (1) platí, že $1234 \equiv 2 \pmod{11}$ a $\mathrm{NSD}(2, 11) = 1$.
Dále $\phi(11) = 10$ a ad (2) platí, že $5678 \equiv 8 \pmod{10}$.
Z předchozího plyne $1234^{5678} \pmod{11} \equiv 2^8 \pmod{11} = 3$. Tj. $1234^{5678} \equiv 3 \pmod{11}$.
▲

Eulerovu větu můžeme použít také pro výpočet multiplikativního inverzního prvku modulo n, a tím i pro řešení lineárních kongruencí $a \cdot x \equiv b \pmod{n}$ (které mají právě jedno řešení právě tehdy když $\mathrm{NSD}(a, n) = 1$.

Lemma 6.14. *Nechť x je multiplikativní inverzní prvek a^{-1} modulo n. Jestliže $NSD(a, n) = 1$ pak*

$$x \equiv a^{-1} \pmod{n}$$

je dáno

$$x \equiv a^{\phi(n)-1} \pmod{n}.$$

Důkaz. Z Eulerovy věty víme, že $a^{\phi(n)} \equiv 1 \pmod{n}$. Pak

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

a $a^{\phi(n)-1}$ je multipikativní inverzní prvek k a modulo n. □

Příklad 6.15. Řešme nejprve kongruenci $5 \cdot x \equiv 14 \pmod{24}$. Protože $\mathrm{NSD}(5, 24) = 1$, má tato kongruence právě jedno řešení. Z předchozího lematu máme

$$x \equiv 14 \cdot 5^{\phi(24)-1} \pmod{24} = 22.$$

▲

Příklad 6.16. Nyní hledáme multipikativní inverzní prvek k 3 modulo 5. Je-li n malé, hledáme $1, \ldots, n-1$ dokud není nalezeno takové $a^{-1} \equiv a^{\phi(n)-1} \pmod{n}$. Takové a^{-1} pro náš příklad je 2, protože $\phi(5) = 4$ a $a^{-1} \equiv 3^{\phi(5)-1} \pmod{5} = 3^3 \pmod{5} = 2$. Ověříme, že platí $a \cdot a^{-1} \equiv 1 \pmod{n}$, tj. $3 \cdot 2 \equiv 1 \pmod{5}$.
▲

Příklady k procvičení

1. Nalezněte $3^{201} \pmod{11}$, použijte Fermatovu větu.
2. Nalezněte $28^{1202} \pmod{13}$, použijte Fermatovu větu.
3. Nalezněte $3^{201} \pmod{11}$, použijte Eulerovu větu.
4. Nalezněte $28^{1202} \pmod{13}$, použijte Eulerovu větu.
5. Pomocí Eulerovy věty najděte multipikativní inverzní prvek k 4 modulo 11.
6. Ukažte na příkladu, že je-li p prvočíslo, pak $a^p \equiv a \pmod{p}$ pro každé c.č. a. Poznámka: uvažte dva případy
   a) $\mathrm{NSD}(a, p) = 1$,
   b) $\mathrm{NSD}(a, p) > 1$, kde $p \mid a$.
7. Vyřešte kongruenci $3 \cdot x \equiv 4 \pmod{29}$.
8. Nechť $n = 7$. Ukažte, že $\{x, x+3, x+3^2, x+3^3, x+3^4, x+3^5, x+3^6\}$ je je úplný systém zbytků modulo 7 pro $x \in \mathbb{Z}$. Nápověda: určete všechny mocniny čísla 3 modulo 7.

6.3 Čínská věta o zbytcích

Algoritmus RSA pracuje s velkými celými čísly, které mohou mít řádově stovky bitů, a proto je např. šifrování pomocí RSA pomalejší než šifrování pomocí některého z symetrických algoritmů. Například symetrický algoritmus AES používá klíč délky 128 bitů. Existuje mnoho způsobů jak urychlit RSA šifrování resp. dešifrování. Jedním z nich je nahradit při dešifrování řešení jedné kongruence řešením soustavy kongruencí, které lze řešit souběžně.

Teoretický předpoklad pro toto urychlení nám poskytuje Čínská věta o zbytcích (Chinese Reminder Theorem, CRT) a algoritmy pro řešení soustav kongruencí, z nichž si jeden ukážeme. Tato věta získala své pojmenování díky čínskému matematikovi Sun Tsu, který žil kolem roku 100 n. l. Větu uvádíme bez důkazu.

Věta 6.17 (Čínská věta o zbytcích). *Nechť $m_1, \ldots, m_k$ jsou po dvou nesoudělná celá čísla, tj. $\forall i, j, (1 \leqq i < j \leqq k) \mid NSD(m_i, m_j) = 1$. Nechť $a_1, \ldots, a_k \in \mathbb{Z}$. Potom soustava kongruencí*

$$\begin{aligned} x &\equiv a_1 \pmod{m_1} \\ x &\equiv a_2 \pmod{m_2} \\ &\ldots\ldots\ldots\ldots \\ x &\equiv a_k \pmod{m_k} \end{aligned}$$

má řešení. Navíc všechna řešení této soustavy kongruencí jsou navzájem kongruentní modulo $M = m_1 \cdot m_2 \cdot \ldots \cdot m_k$.

Čínská věta říká, že libovolné číslo z množiny $\{0, 1, \ldots, M\}$ (tj. zbytkovou třídu modulo M) lze jednoznačným způsobem reprezentovat pomocí zbytkových tříd modulo $m_1, \ldots, m_k$. Odtud vyplývá následující tvrzení.

Lemma 6.18. *Nechť p, q jsou navzájem nesoudělná celá čísla. Potom pro libovolné $x \in Z_{p \cdot q}$ platí:*

$$x \equiv a \pmod{p \cdot q} \Leftrightarrow x \equiv a \pmod{p}, x \equiv a \pmod{q}.$$

Gaussův algoritmus: Řešení x kongruencí uvedených v CRT 6.17 se může počítat jako

$$x \equiv \sum_{i=1}^{k} a_i N_i L_i \pmod{M},$$

kde $M = m_1 m_2 \ldots m_k$, $N_i = M/m_i$ a $L_i = N_i^{-1} \pmod{m_i}$. Uvedený výpočet může být proveden se složitostí $O((log\ n)^2)$ bitových operací.

Příklad 6.19. Uvažujme soustavu kongruencí

$$\begin{aligned} x &\equiv 3 \pmod{6}, \\ x &\equiv 2 \pmod{7}. \end{aligned}$$

Řešením je $x = 3 \cdot 7 \cdot 1 + 2 \cdot 6 \cdot 6 \pmod{42} = 93 \pmod{42} = 9$. Nejmenší celé číslo, které je řešením soustavy je $93 \pmod{(6 \cdot 7)} = 9$.
▲

Ukážeme si, jak můžeme CRT využít pro sčítání a násobení modulo M. Nechť $M = m_1 m_2 \dots m_k$. Každé celé číslo ze $\mathbb{Z}_M$ můžeme reprezentovat jako k-tici prvků ze $\mathbb{Z}_{m_i}$ jako $A \leftrightarrow (a_1, a_2, \dots, a_k)$, kde $A \in \mathbb{Z}_M, a_i \in \mathbb{Z}_{m_i}$ a $a_i \equiv A \pmod{m_i}$ pro $1 \leqq i \leqq k$. Operace v $\mathbb{Z}_M$ mohou být prováděny ekvivalentně s korespondující k-ticí. Např. mějme čísla $A \leftrightarrow (a_1, a_2, \dots, a_k)$, $B \leftrightarrow (b_1, b_2, \dots, b_k)$. Součet a násobení čísel A a B můžeme realizovat díky CRT takto:

$$\begin{gathered}(A + B) \pmod{M} \leftrightarrow ((a_1 + b_1) \pmod{m_1}, (a_2 + b_2) \pmod{m_2}, \dots \\ \dots, (a_k + b_k) \pmod{m_k}), \\ (A \cdot B) \pmod{M} \leftrightarrow ((a_1 \cdot b_1) \pmod{m_1}, (a_2 \cdot b_2) \pmod{m_2}, \dots \\ \dots, (a_k \cdot b_k) \pmod{m_k}).\end{gathered}$$

Příklad 6.20. Nechť $M = 1813 = 37 \cdot 49 = m_1 \cdot m_2$. Mějme číslo $A = 973$, pak podle Gaussova algoritmu

$$N_1 = 1813/37 = 49, N_2 = 1813/49 = 37.$$

Dále

$$\begin{aligned}L_1 = 34 \pmod{37},\ N_1^{-1} \pmod{m_1} = 49^{-1} \pmod{37} = 34, \\ L_2 = 4 \pmod{49},\ N_2^{-1} \pmod{m_2} = 37^{-1} \pmod{49} = 4.\end{aligned}$$

Nechť $a_i \equiv A \pmod{m_i}$. Pak

$$\begin{aligned}a_1 \equiv A \pmod{m_1} = 973 \pmod{37} = 11, \\ a_2 \equiv A \pmod{m_2} = 973 \pmod{49} = 42.\end{aligned}$$

Chceme sečíst $(A + B) \pmod{M} = (973 + 678) \pmod{1813}$

$$\begin{aligned}b_1 \equiv B \pmod{m_1} = 678 \pmod{37} = 12, \\ b_2 \equiv B \pmod{m_2} = 678 \pmod{49} = 41.\end{aligned}$$

Dále

$$\begin{aligned}x \equiv (a_1 + b_1) \pmod{m_1} = (11 + 12) \pmod{37} = 23, \\ x \equiv (a_2 + b_2) \pmod{m_2} = (42 + 41) \pmod{49} = 34.\end{aligned}$$

Podle Gaussova algoritmu

$$\begin{gathered}x \equiv 23 \pmod{37} \text{ a } x \equiv 34 \pmod{49}, \\ \text{tj. } x = 23 \cdot 49 \cdot 34 + 34 \cdot 37 \cdot 4 \pmod{1813} = 43350 \pmod{1813} = 1651.\end{gathered}$$

Kontrola: $(973 + 678) \pmod{1813} = 1651$.

Nyní chceme vynásobit $1651 \pmod{1813}$ a 73. Můžeme násobit $(23, 34)$ číslem 73 a provést modulární redukci, pak obdržíme $(23 \cdot 73 \pmod{37}, 34 \cdot 73 \pmod{49} = (14, 32)$. Snadno ověříme, že

$$14 \cdot 49 \cdot 34 + 32 \cdot 37 \cdot 4 \pmod{1813} = 865$$

Kontrola: $1651 \cdot 73 \pmod{1813} = 865$.
▲

Příklady k procvičení

1. Vyřešte soustavu kongruencí:
   $x \equiv 2 \pmod{3}$,
   $x \equiv 3 \pmod{5}$,
   $x \equiv 2 \pmod{7}$.

2. Vyřešte soustavu kongruencí:
   $x \equiv 2 \pmod{7}$,
   $x \equiv 7 \pmod{9}$,
   $x \equiv 3 \pmod{4}$.

3. Nechť $M = 1001$, pak $m_1 = 7, m_2 = 11$ a $m_3 = 13$. Vyřešte soustavu kongruencí:
   $x \equiv 5 \pmod{7}$,
   $x \equiv 3 \pmod{11}$,
   $x \equiv 10 \pmod{13}$.

4. Šesti profesorům začínají kurzy v pondělí, v úterý, ve středu, ve čtvrtek, v pátek resp. v sobotu. Oznámili, že mají v úmyslu přednášet v intervalech $2, 3, 4, 1, 6$ resp. 5 dní. Nicméně předpisy univerzity zakazují výuku v neděli (takže profesoři musí vynechat přednášku v neděli). Kdy nejdříve bude muset každý ze šesti profesorů vynechat přednášku (protože v neděli učit nesmí)? Použijte CRT.

5. Žena na trhu prodává vejce. Má jich plný košík. Kolemjdoucí však byl neopatrný, zavadil o košík a všechna vejce se rozbila. Nabídl ženě, že jí za rozbitá vejce zaplatí. Ale kolik jich bylo? To žena nevěděla. Věděla pouze, že v případě, že by se vejce počítala po dvou, jedno by zbylo. Počítala-li by se po třech, zbyla by dvě; počítala-li by se po pěti, tři by zbyla; a nakonec počítala-li by se po sedmi, zbyla by čtyři vejce. Kolik vajec měla žena v košíku? Opět použijte CRT.

6.4 Problém diskrétního logaritmu

V této části si představíme další dva pojmy důležité pro kryptografii. Prvním z nich je pojem řád prvku, druhým diskrétní logaritmus.

Definice 6.21. Nechť n je kladné celé číslo a nechť a je takové celé číslo, pro které platí $\mathrm{NSD}(a, n) = 1$. Potom *řád a modulo n* je nejmenší m takové, že $a^m \equiv 1 \pmod{n}$, budeme ho označovat $ord_n(a)$.

Pojem řád a modulo n je moderní pojem z algebry, viz poznámka 3.10. Starší pojem (užívaný Gaussem), se kterým se můžeme setkat zejména v anglické terminologii, je „exponent, kterému a náleží $\pmod{m}$“ (a belongs to exponent m). Dále se číslo označuje také jako *délka periody*, kterou generuje a.

Příklad 6.22. Jaký je řád celého čísla 7 (mod 19)?
$7^1 \equiv 7 \pmod{19}$,
$7^2 = 49 = 2 \cdot 19 + 11 \equiv 11 \pmod{19}$,
$7^3 = 343 = 18 \cdot 19 + 1 \equiv 1 \pmod{19}$
$7^4 = 2401 = 126 \cdot 19 + 7 \equiv 7 \pmod{19}$,
$7^5 = 16807 = 884 + 19 + 11 \equiv 11 \pmod{19}$.
Nemá smysl dále pokračovat, neboť posloupnost je periodická a délka periody je nejmenší exponent m takový, že $7^m \equiv 1 \pmod{19}$. Tedy řád prvku 7 (mod 19) je $m = 3$.
▲

Příklad 6.23. Tabulka 6.1 reprezentuje hodnoty $a^i \pmod{11}$ pro $i = 1, 2, \ldots, 10$. Z této tabulky obdržíme
$ord_{11}(1) = 1$,
$ord_{11}(10) = 2$,
$ord_{11}(3) = ord_{11}(4) = ord_{11}(5) = ord_{11}(9) = 5$,
$ord_{11}(2) = ord_{11}(6) = ord_{11}(7) = ord_{11}(8) = 10$.
▲

Lemma 6.24. *Nechť n je kladné celé číslo, a nechť a je takové celé číslo, pro které platí $NSD(a, n) = 1$ a nechť $m = ord_n(a)$. Pak*

a) Jestliže $a^r \equiv 1 \pmod{n}$, pak $m \mid r$.

b) $m \mid \phi(n)$.

c) Pro celá čísla s a t platí $a^s \equiv a^t \pmod{n}$ právě tehdy když $s \equiv t \pmod{m}$.

d) žádná dvě čísla $a, a^2, \ldots, a^m$ nejsou kongruentní modulo n.

e) Jestliže r je kladné c.č., pak řád a^r modulo n je $\frac{m}{NSD(m,r)}$.

f) Řád a^r modulo n je m právě tehdy když $NSD(r, m) = 1$.

a	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^{10}
1	1	1	1	1	1	1	1	1	1
2	4	8	5	10	9	7	3	6	1
3	9	5	4	1	3	9	5	4	1
4	5	9	3	1	4	5	9	3	1
5	3	4	9	1	5	3	4	9	1
6	3	7	9	10	5	8	4	2	1
7	5	2	3	10	4	6	9	8	1
8	9	6	4	10	3	2	5	7	1
9	4	3	5	1	9	4	3	5	1
10	1	10	1	10	1	10	1	10	1

Tab. 6.1 Hodnoty $a^i \pmod{11}$ pro $i = 1, 2, \ldots, 10$

Poznámka 6.25. Vztah mezi algebrou (teorií grup) a teorií čísel vyplývá z věty 3.8. Jestliže a je prvek grupy G, potom řád prvku a dělí řád grupy G. Pokud se podíváme na příklad 6.23, vidíme, že pracujeme s grupou $\mathbb{Z}_{11}^*$ a například řád prvku $3, ord_{11}(3) = 5$, určitě dělí řád této grupy, který je $|\mathbb{Z}_{11}^*| = 10 = \phi(11)$.

Definice 6.26. Nechť a, n jsou nesoudělná celá čísla (n je kladné celé číslo). Jestliže řád c.č. a modulo n je $\phi(n)$, tj. $ord_n(a) = \phi(n)$, pak se a nazývá *primitivní odmocnina* čísla n.

V tabulce 6.2 vidíme mocniny $a < 19$ modulo 19. Všechny posloupnosti končí jedničkou a řád každého prvku (délka posloupností) dělí $\phi(19) = 18$. Některé z posloupností mají délku 18, tj. některé prvky mají řád $\phi(19) = 18$. Jsou to např. prvky $a = 2, a = 3$. Vidíme, že pak mocniny takového c.č., pro které $ord_n(a) = \phi(n)$ (například $ord_{19}(2) = \phi(19) = 18$), generují všechna c.č. od 1 do $n - 1$, tj. čísla $a \pmod{n}, a^2 \pmod{n}, \ldots, a^{n-1} \pmod{n}$ (všechna c.č. od 1 do $n - 1$ v nějaké permutaci).

Poznámka 6.27. Doplňme nyní další poznatky o primitivní odmocnině.

1. Předpokládejme, že $\mathrm{NSD}(a, n) = 1$. Jestliže je a primitivní odmocnina modulo n, pak množina $\{a, a^2, \ldots, a^{\phi(n)}\}$ tvoří redukovaný systém zbytků modulo n.

2. Jestliže p je prvočíslo, potom existuje $\phi(p-1)$ (nekongruentních) primitivních odmocnin modulo p.

3. Jestliže n má primitivní odmocninu, potom existuje $\phi(\phi(n-1))$ (nekongruentních) primitivních odmocnin modulo n.

4. Celé číslo $n > 1$ má primitivní odmocninu modulo n jestliže $n = 2, 4, p^\alpha$ nebo $2p^\alpha$, kde p je liché prvočíslo a α je kladné celé číslo.

5. Jestliže $n = 2^\alpha$, kde $\alpha \geqq 3$ nebo $n = 2^\alpha p_1^{\alpha_1} \dots p_k^{\alpha_k}$, kde $\alpha \geqq 2$ nebo $k \geqq 2$, pak neexistují žádné primitivní odmocniny modulo n.

Příklad 6.28. Nechť $n = 34$, pak existuje $\phi(\phi(34)) = 8$ primitivních odmocnin modulo 34. Přesněji to jsou $3, 5, 7, 11, 23, 27, 29, 31$. Nyní mějme primitivní odmocninu $a = 5$, $\text{NSD}(5, 34) = 1$, $\phi(34) = 16$. Pak

$$\{a, a^2, \dots, a^{\phi(n)}\} = \{5^1, 5^2, \dots, 5^{\phi(34)}\} = \{5^1, 5^2, \dots, 5^{16}\} =$$

$$= \{5, 25, 23, 13, 31, 19, 27, 33, 29, 9, 11, 21, 3, 15, 7, 1\} =$$

$$= \{1, 3, 5, 7, 9, 11, 13, 15, 19, 21, 23, 25, 27, 29, 31, 33\},$$

což je redukovaný systém zbytků modulo 34.
▲

Příklad 6.29. Nechť $n = 11$, pak $\phi(11) = 10$ a $\phi(11 - 1) = \phi(10) = 4$, existují tedy 4 primitivní odmocniny modulo 11 a to jsou $2, 6, 7, 8$. Ověřte (a porovnejte s příkladem 6.23).
▲

Postupně jsme se dostali k velmi důležitému pojmu diskrétního logaritmu. Předpokládejme vztah $b \equiv a^i \pmod p$. Je-li dáno i, a a prvočíslo p, není obtížné vypočítat $b \equiv a^i \pmod p$. Je-li však dáno prvočíslo p, generátor a cyklické grupy $\mathbb{Z}_p^*$ a $b \in \mathbb{Z}_p^*$, je výpočetně obtížné nalézt i takové že $b \equiv a^i \pmod p$, $i \in \langle 1, p - 1\rangle$, tedy vyřešit tzv. *problém diskrétního logaritmu* (DLP).

Jeden z nejrychlejších algoritmů pro nalezení diskrétního logartimu General Number Field Sieve má subexponenciální asymptotickou složitost $e^{((\ln p)^{1/3} \ln(\ln p))^{2/3}}$, což není pro velká p řešitelné. Na problému diskrétního logaritmu je založena celá řada kryptografických algoritmů, z nichž nejznámější je algoritmus Diffie - Hellman, algoritmus ElGamal nebo algoritmy založené na úloze diskrétního logaritmu pro eliptické křivky (viz kapitola 7).

Koncept diskrétního logaritmu (nebo také indexu celého čísla) modulo n poprvé představil C. F. Gauss v Disquisitiones Arithmeticae. Má-li celé číslo n primitivní odmocninu a modulo n, pak množina $\{a, a^2, \dots, a^{\phi(n)}\}$ tvoří redukovaný systém zbytků modulo n. Zároveň je a generátor cyklické grupy redukovaných zbytků modulo n (viz kapitola 3.1). Tedy je-li $\text{NSD}(b, n) = 1$, pak se b může vyjádřit ve tvaru $b \equiv a^i \pmod n$ pro $1 \leqq i \leqq \phi(n)$. Odtud už se dostáváme k následující definici diskrétního logaritmu a k jeho příkladu (viz tabulka 6.3).

Definice 6.30. Nechť a je primitivní odmocninu c. čísla $n > 1$. Jestliže $\text{NSD}(b, n) = 1$, pak nejmenší kladné celé číslo i pro které $b \equiv a^i \pmod n$, kde $1 \leqq i \leqq \phi(n)$, se nazývá *diskrétní logaritmus* čísla b o základu $a \pmod n$

Lemma 6.31. *Jestliže a je primitivní odmocnina modulo n, pak pro celá čísla s a t $a^s \equiv a^t \pmod{n}$ právě tehdy když $s \equiv t \pmod{\phi(n)}$.*

Poznámka 6.32. Vlastnosti diskrétního logaritmu jsou podobné vlastnostem logaritmu reálného čísla o daném základu. Když si diskrétní logaritmus i vyjádříme jako $log_a\, b$, dostáváme $b \equiv a^{log_a\, b} \pmod{n}$ pro $1 \leqq i \leqq \phi(n)$, jak ukazují následující poznatky:

a) Nechť a je primitivní odmocnina modulo prvočíslo p a $\mathrm{NSD}(b, p) = 1$. Pak $a^k \equiv b \pmod{p}$ právě tehdy když $k \equiv log_a\, b \pmod{(p-1)}$.

b) Nechť n je kladné celé číslo s primitivní odmocninou a, a nechť $\mathrm{NSD}(b, n) = \mathrm{NSD}(c, n) = 1$. Pak

   i) $log_a\, 1 \equiv 0 \pmod{\phi(n)}$.

   ii) $log_a\, (bc) \equiv log_a\, b + log_a\, c \pmod{\phi(n)}$.

   iii) $log_a\, b^k \equiv k \cdot log_a\, b \pmod{\phi(n)}$, jestliže k je kladné celé číslo.

Příklad 6.33. Hledáme řešení kongruence $15 \equiv 6^i \pmod{109}$, tedy řešíme problém diskrétního logaritmu pro $b = 15, a = 6, n = 109$. Budeme postupně zkoušet $k = 1, 2, 3, \ldots$ dokud nenalezneme takové k které vyhovuje $6^k \pmod{109} = 15$:

$6^1 \equiv 6 \pmod{109}$ $\quad$ $6^{11} \equiv 39 \pmod{109}$
$6^2 \equiv 36 \pmod{109}$ $\quad$ $6^{12} \equiv 16 \pmod{109}$
$6^3 \equiv 107 \pmod{109}$ $\quad$ $6^{13} \equiv 96 \pmod{109}$
$6^4 \equiv 97 \pmod{109}$ $\quad$ $6^{14} \equiv 31 \pmod{109}$
$6^5 \equiv 37 \pmod{109}$ $\quad$ $6^{15} \equiv 77 \pmod{109}$
$6^6 \equiv 4 \pmod{109}$ $\quad$ $6^{16} \equiv 26 \pmod{109}$
$6^7 \equiv 24 \pmod{109}$ $\quad$ $6^{17} \equiv 47 \pmod{109}$
$6^8 \equiv 35 \pmod{109}$ $\quad$ $6^{18} \equiv 64 \pmod{109}$
$6^9 \equiv 101 \pmod{109}$ $\quad$ $6^{19} \equiv 57 \pmod{109}$
$6^{10} \equiv 61 \pmod{109}$ $\quad$ $6^{20} \equiv 15 \pmod{109}$

Nejmenší k, takové že $6^k \pmod{109} = 15$ je $k = 20$, tedy $15 \equiv 6^{20} \pmod{109}$.
▲

Příklady k procvičení

1. Jaký je řád $3, 5, 7$ modulo 8?

2. Ověřte,zda je či není 17 primitivní odmocnina modulo 45.

3. Ověřte, zda je či není 7 primitivní odmocnina modulo 46.

4. Ukažte, že 11 je primitivní odmocnina modulo 31.

5. Jaké jsou primitivní odmocniny modulo 47 a kolik jich je?

6. Nalezněte všechny primitivní odmocniny modulo 25.

7. Nalezněte primitivní mocniny a modulo 17 a diskrétní logaritmy modulo 17.

8. Nalezněte primitivní mocniny a modulo 18 a diskrétní logaritmy modulo 18.

9. Nalezněte diskrétní logaritmy x (pozor, kongruence nemusí mít řešení, proč?) pro:
   - $3 \equiv 2^x \pmod 5$,
   - $3 \equiv 4^x \pmod 7$,
   - $2 \equiv 4^x \pmod{13}$,
   - $3 \equiv 5^x \pmod 7$,
   - $10 \equiv 6^x \pmod{11}$,
   - $3 \equiv 7^x \pmod{13}$,
   - $3 \equiv 8^x \pmod{11}$,
   - $6 \equiv 9^x \pmod{11}$.

10. Mějme 2 jako primitivní odmocninu modulo 29. Sestavte tabulku diskrétních logaritmů modulo 29 a použijte ji pro řešení následujících kongruencí:
    - $17x^2 \equiv 10 \pmod{29}$,
    - $x^2 - 4x - 16 \equiv 0 \pmod{29}$,
    - $x^2 \equiv 17 \pmod{13}$.

a	a^2	a^3	a^4	a^5	a^6	a^7	a^8	a^9	a^{10}	a^{11}	a^{12}	a^{13}	a^{14}	a^{15}	a^{16}	a^{17}	a^{18}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
3	9	8	5	15	7	2	6	18	16	10	11	14	4	12	17	13	1
4	16	7	9	17	11	6	5	1	4	16	7	9	17	11	6	5	1
5	6	11	17	9	7	16	4	1	5	6	11	17	9	7	16	4	1
6	17	7	4	5	11	9	16	1	6	17	7	4	5	11	9	16	1
7	11	1	7	11	1	7	11	1	7	11	1	7	11	1	7	11	1
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1

Tab. 6.2 Mocniny modulo 19

b	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
i $(a = 2)$	18	1	13	2	16	14	6	3	8	17	12	15	5	7	11	4	10	9
b	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
i $(a = 3)$	18	7	1	14	4	8	6	3	2	11	12	15	17	13	5	10	16	9

Tab. 6.3 Tabulka některých diskrétních logaritmů modulo 19

Kapitola 7

Kryptografie na bázi eliptických křivek

Dnešní systémy s veřejným klíčem (asymetrické kryptosystémy) vznikly proto, aby byl s jejich pomocí vyřešen problém klíčového hospodářství pro symetrické algoritmy. Pro N účastníků komunikačního procesu potřebujeme $(N*(N-1))/2$ tajných klíčů, které sdílí vždy dvojice účastníků, což sebou pro velká N nese řadu problémů (s distribucí klíčů, jejich aktualizací atd.). Pokud však účastníci používají asymetrický kryptosytém, kdy každý účastník vlastní dvojicí klíčů - veřejný a soukromý, některé problémy odpadají. Nevýhodou asymetrických kryptosystémů je, že jsou pomalejší než symetrické. Je to jednak kvůli jejich výpočetní obtížnosti a pak kvůli bezpečné velikosti klíče, která je v některých případech podstatně větší než u symetrických kryptosystémů, a asymetrické algoritmy. Odborníci proto hledají nové algoritmy pro asymetrické kryptosystémy.

V tabulce 7.1 je uvedena srovnatelná bezpečnost různých kryptosystémů při různých délkách klíčů podle amerického NISTu (National Institute of Standards and Technology). Tabulka z roku 2007 ukazuje doporučené délky klíčů pro různé kryptosystémy s výhledem na několik desítek let. Porovnejme např. délky klíčů ve třetím a posledním sloupci, tedy klíče pro asymetrické kryptosystémy na bázi faktorizace a asymetrické kryptosystémy na bázi eliptických křivek.

Mnoho současných asymetrických kryptosystémů je založeno na využití operací nad algebraickými strukturami. Kryptografická síla těchto systémů je odvozena z výpočetní obtížnosti známých problémů s vysokou složitostí (problém faktorizace velkého celého čísla nebo problém diskrétního logaritmu). Kryptografie na bázi eliptických křivek (Elliptic Curve Cryptography, dále jen ECC) je moderní směr, který v řadě ukazatelů přináší lepší výsledky než nejrozšířenější používané kryptosystémy.

Eliptické křivky byly zkoumány již více než jedno století zejména kvůli jejich matematické zajímavosti. V současné době jsou však využívány v mnoha aplikacích. Zmiňme algoritmy pro faktorizaci celého čísla, pro testování prvočíselnosti a rovněž

Datum	Symetrický klíč	Asymetrický klíč RSA	Diskrétní logaritmus		Eliptická křivka
			klíč	grupa	
2007 - 2010	80	1024	160	1024	160
2011 - 2030	112	2048	224	2048	224
$>$ 2030	128	3072	256	3072	256
$\gg$ 2030	192	7680	384	7680	384
$\ggg$ 2030	256	15360	512	15360	512

Tab. 7.1 Srovnatelná bezpečnost podle NIST

aplikace kryptografické. Užití eliptických křivek pro návrh asymetrických kryptosystémů poprvé navrhli v r. 1985 nezávisle pánové Victor Miller a Neal Koblitz. Jedná se vlastně o analogii již existujících systémů s veřejným klíčem, kdy je modulární aritmetika nahrazena aritmetikou budovanou na základě operací s body na eliptické křivce. U asymetrických kryptosystémů definovaných nad eliptickou křivkou (ECC) se hierarchicky volí dva typy algebraických struktur: konečné těleso a eliptická křivka reprezentující grupu bodů, nad níž je vlastní asymetrický algoritmus definován. Volba obou těchto algebraických struktur významně ovlivňuje bezpečnost a efektivitu kryptosystému. Požadavky kladené na tyto dvě struktury spolu vzájemně souvisí.

Bezpečnost eliptických kryptosystémů spočívá v obtížnosti řešení úlohy diskrétního logaritmu pro eliptické křivky. V současné době je tato úloha podstatně obtížněji řešitelná než je úloha klasického diskrétního logaritmu. Dokonce nejsou pro tyto algoritmy známy žádné subexponenciální algoritmy (jako např. pro klasický diskrétní logaritmus, viz kap. 6.4), nejlepší algoritmy mají plně exponenciální charakter. V důsledku toho lze konstruovat bezpečné kryptosystémy s výrazně kratší délkou klíče (viz tabulka 7.1). To vede mimo jiné k implementacím s menšími nároky na paměť, které jsou současně i výrazně rychlejší ve srovnání např. s kryptosystémy na bázi diskrétního logaritmu. Teoretická konstrukce přitom umožňuje vytvořit systémy zcela analogické klasickým modelům.

7.1 Pojem eliptické křivky

Pod pojmem rovinná křivka rozumíme množinu bodů, které splňují rovnici $F(x, y) = 0$. Nejjednodušší rovinnou křivkou je přímka. Rovnici přímky lze zapsat jako polynom s proměnnými x a y, přitom tyto proměnné se v rovnicí objevují v podobě lineární závislosti. Obecná rovnice přímky má tvar $ax + by + c = 0$, přičemž $a \neq 0$ nebo $b \neq 0$ a x, y jsou souřadnice libovolného bodu přímky. Kuželosečky (parabola, hyperbola, elipsa) lze popsat rovnicemi, kde závislost proměnných je popsána kvadratickou rovnicí (v x a y). Logicky navazují kubické křivky, závislost proměnných

je popsána rovnicí třetího stupně. Jejich speciální podtřídou jsou eliptické křivky.

Eliptická křivka je algebraická struktura konstruovaná obecně nad tělesem (např. reálných čísel). Eliptická křivka je hladká spojitá křivka, na které definujeme bod $\mathcal{O}$, což je bod v nekonečnu. Eliptická křivka E definovaná nad konečným tělesem F (v literatuře označováno jako $E|F$), kde $a_1, a_2, a_3, a_4, a_5 \in F$, je reprezentovaná pomocí tzv. Weierstrassovy rovnice:

$$E: \; y^2 + a_1xy + a_2y = x^3 + a_3x^2 + a_4x + a_5.$$

Pro naše účely můžeme eliptickou křivku nad reálnými čísly definovat jednodušeji, jako skupinu souřadnic (x, y), které vyhovují rovnici $y^2 = x^3 + ax + b$, kde $a, b, x, y \in \mathbb{R}$, což je jedna ze zjednodušených forem Weierstrassovy rovnice. Pokud je daná eliptická křivka nesingulární, může zformovat grupu. Grupy eliptických křivek jsou aditivní grupy, to znamená, že základní operací je zde sčítání. Sčítání dvou bodů na eliptické křivce E je definováno geometricky. Součtem dvou různých bodů P, Q, které na křivce leží, je opět bod křivky E, což si postupně vysvětlíme.

Protože pracujeme v rovině, jsou souřadnicemi bodů na křivce (x, y) reálná čísla. Reálná čísla jsou tělesem, avšak pro kryptografické účely budeme, jako i v jiných případech, používat jiná tělesa. Pro počítačové zpracování je nejpřirozenější pracovat s m-ticemi bitů. To nám umožní Galoisovo těleso $\mathrm{GF}(2^m)$, které obsahuje právě všechny m-tice bitů. Dalším vhodným kandidátem je Galoisovo těleso $\mathrm{GF}(p)$, kde p je prvočíslo. Toto těleso obsahuje čísla $\{0, 1, \ldots, p-1\}$, kde p je obvykle velmi velké prvočíslo, a výpočty v něm se provádějí modulo p, viz kapitola 3.3. V kryptografii se tedy používají eliptické křivky nad konečnými tělesy, která lze algebraicky klasifikovat, a každé konečné těleso je pak jednoznačně určeno svým řádem (počtem svých prvků).

Definice 7.1. Existuje-li kladné celé číslo n takové, že v tělese F platí

$$\underbrace{1 + 1 + \ldots + 1}_{n} = 0$$

pak nejmenší takové kladné číslo nazýváme *charakteristika tělesa* F. Pokud žádné takové kladné celé číslo n neexistuje, tak říkáme, že těleso F má charakteristiku 0.

Příklad 7.2. Reálná čísla $\mathbb{R}$ a racionální čísla $\mathbb{Q}$ s operacemi sčítání a násobení tvoří tělesa s charakteristikou 0. Celá čísla modulo prvočíslo p, $\mathbb{Z}_p$, jsou tělesa s charakteristikou p. Galoisova tělesa $GF(q)$, kde $q = p^m$, jsou tělesa s charakteristikou p, kde p je prvočíslo.
▲

Mějme nyní je konečné těleso GF_q, kde $q = p^m$ je počet prvků, p prvočíslo a m přirozené číslo. Toto těleso budeme dále označovat F_q. Definujme eliptickou křivku nad konečným tělesem F_q.

Definice 7.3. *Eliptická křivka* nad konečným tělesem F_q je množina

$$E = \{(x, y) \in F_q^2 \setminus \{[0, 0]\}, F(x, y) = 0\} \cup \{\mathcal{O}\},$$

kde $\mathcal{O}$ je dodefinovaný neutrální prvek eliptické křivky (tzv. bod v nekonečnu) a

$$F(x, y) = y^2 + a_1xy + a_2y - x^3 - a_3x^2 - a_4x - a_5$$

je polynom nad F_q.

Na takto vytvořené množině je možno definovat binární operaci (sčítání bodů) tak, že eliptická křivka opatřená touto operací má strukturu Abelovské (komutativní) grupy. Vztah $F(x, y) = 0$ je obecnou Weierstrassovou rovnicí eliptické křivky. Tu lze zjednodušit pro jednotlivé tvary těles a doplnit podmínky pro hodnoty koeficientů rovnic tak, aby eliptická křivka nebyla singulární (aby neexistoval tzv. bod singularity), tj. aby mohla daná eliptická křivka tvořit grupu nad tělesem F_q.

1. Pro $q = p^m$, kde $p > 3$ (tj. máme těleso charakteristiky > 3), $m \geqq 1$ a koeficienty $a, b \in F_q$, $4a^3 + 27b^2 \not\equiv 0$, splňuje *bod eliptické křivky* $(x, y) \in E \setminus \{\mathcal{O}\}$ následující rovnici nad F_q:

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

   U prvočíselných polí s charakteristikou $p > 3$ se provádí následující transformace Weierstrassovy rovnice, viz [3], abychom obdrželi uvedenou rovnici:

   $$(x, y) \to \left(\frac{x - 3a_1^2 - 12a_2}{36}, \frac{y - 3a_1x}{216} - \frac{a_1^3 + 4a_1a_2 - 12a_3}{24}\right)$$

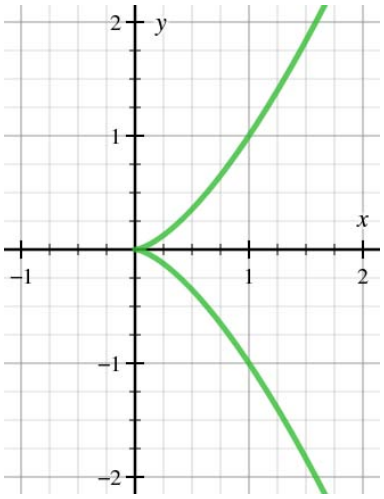
2. Pro $q = 2^m$ (tj. máme těleso charakteristiky 2), $m \geqq 1$ a koeficienty $a, b \in F_q$, $b \neq 0$, splňuje bod eliptické křivky $(x, y) \in E \setminus \{\mathcal{O}\}$ následující rovnici nad F_q:

   $$y^2 + xy = x^3 + ax^2 + b.$$

3. Pro $q = 3^m$ (tj. máme těleso charakteristiky 3), $m \geqq 1$ a koeficienty $a, b \in F_q$, $b \neq 0$, splňuje bod eliptické křivky $(x, y) \in E \setminus \{\mathcal{O}\}$ následující rovnici nad F_q:

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

Doplňme pouze obecnou formulaci pojmu singularita. V geometrii se jako singularita (singulární bod) označuje takový bod křivky (plochy nebo prostoru), v němž není křivka (plocha, prostor) lokálně hladká, viz obrázek 7.1. V opačném případě se jedná o regulární bod. Regulární bod křivky je takový bod, v němž existuje jediná tečna, v ostatních případech je bod singulární.



Obr. 7.1 Singulární křivka $E : x^3 = y^2$

7.2 Operace sčítání bodů

Sčítání dvou bodů na eliptické křivce můžeme definovat geometricky i algebraicky. Nejprve přibližme operaci sčítání na *reálné* křivce (v rovině) geometricky:

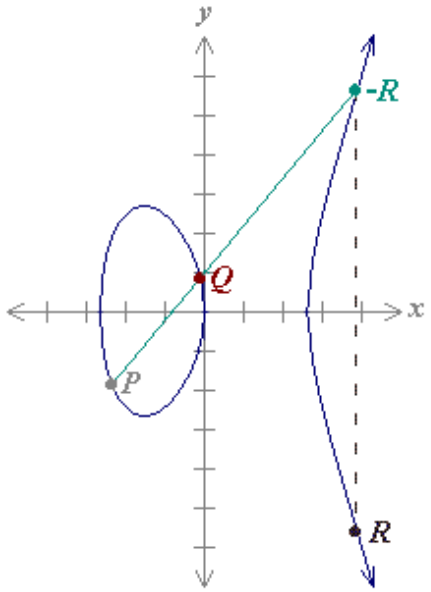
- Výsledkem součtu $P + Q$ dvou různých bodů P, Q, které leží na křivce E, bude opět bod R křivky E a vznikne takto: Spojíme body $P = (x_P, y_P)$ a $Q = (x_Q, y_Q)$ přímkou, ta protne křivku v dalším bodě, který označíme $-R$, a výsledkem sčítání je bod R, symetrický k $-R$ podle osy x, viz obr. 7.2. Body symetrické podle osy x se nazývají *opačné*.

- Když sčítáme body opačné $P + (-P)$, jejich spojnice (rovnoběžka s osou y) eliptickou křivku E protne jakoby v nekonečnu. Proto matematici definitoricky přidali ke křivce E bod v nekonečnu $\mathcal{O}$ a sčítání dodefinovali i pro body opačné: $P + (-P) = \mathcal{O}$, viz obr. 7.3. Bod v nekonečnu si můžeme představit jako bod ležící daleko ve směru osy y.

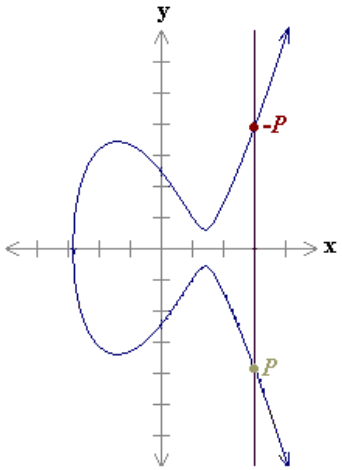
- Pro bod v nekonečnu $\mathcal{O}$ definujeme pravidla pro sčítání takto: pro každý bod P na křivce E definujeme $P + \mathcal{O} = P$ a také $\mathcal{O} + \mathcal{O} = \mathcal{O}$, přičemž $-\mathcal{O} = \mathcal{O}$.

- Pro přičtení bodu k sobě samému, tj. $P + P$, je vedena tečna ke křivce E z bodu P. Jestliže $y_P \neq 0$, protne tečna křivku v bodě $-R$. Výsledkem sčítání $P + P = 2P = R$ je bod R, symetrický k $-R$ podle osy x, viz obr. 7.4. Pro $y_P = 0$ je $P + P = 2P = \mathcal{O}$, viz obr. 7.5.

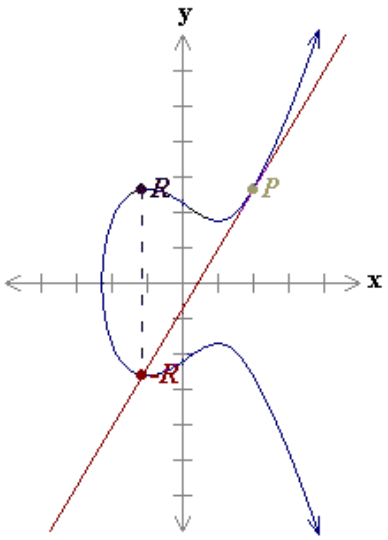
Body eliptické křivky E spolu s operací sčítání tvoří, jak už jsme zmínili, Abelovskou grupu, kde $\mathcal{O}$ je neutrální (nulový prvek). Ověříme nyní, že tomu tak je, na geometrické interpretaci operace sčítání bodů:



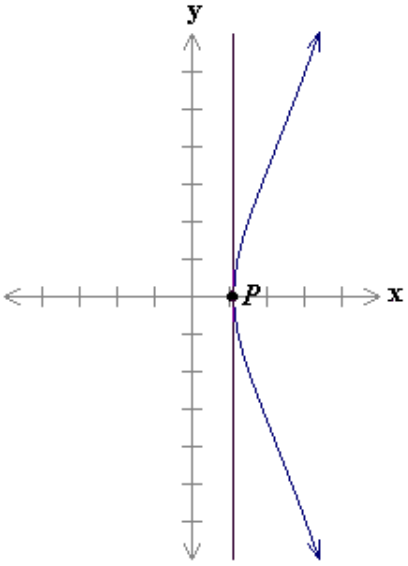
Obr. 7.2 Sčítání $P + Q = R$ bodů křivky $E : y^2 = x^3 - 7x$



Obr. 7.3 Sčítání $P + (-P) = \mathcal{O}$ bodů křivky $E : y^2 = x^3 - 6x + 6$



Obr. 7.4 Sčítání $P + P = 2P$ bodů křivky $E : y^2 = x^3 - 3x + 5$



Obr. 7.5 Sčítání $P + P = 2P = \mathcal{O}$ bodů křivky $E : y^2 = x^3 + 5x - 7$, $y_P = 0$

- Jestliže přímka l protíná křivku E v (ne nutně různých) bodech P, Q a R pak $(P + Q) + R = \mathcal{O}$.
- $P + \mathcal{O} = P$ pro $\forall P \in E$.
- $P + Q = Q + P$ pro $\forall P, Q \in E$.
- $P + Q = Q + P$ pro $\forall P, Q \in E$.
- Nechť $P \in E$, pak existuje bod $-P \in E$, takový, že $P + (-P) = \mathcal{O}$.
- Nechť $P, Q, R \in E$, pak $(P + Q) + R = P + (Q + R)$.

Navíc, pokud je eliptická křivka E definovaná nad konečným tělesem F_q ($E|F_q$, viz definice 7.3), je pak $E|F_q$ podgrupou grupy E.

Geometrická interpretace sčítání bodů nám poskytuje jasnou představu o tom, jak mohou být dva body eliptické křivky sečteny, abychom obdrželi bod třetí. Pro vlastní výpočet ale samozřejmě potřebujeme takovéto sčítání definovat algebraicky.

- Algebraicky je směrnice přímky, která spojuje body P, Q (předpokládejme, že jsou různé a nikoli opačné), rovna $s = (y_Q - y_P)/(x_Q - x_P)$ a souřadnice bodu $R = (x_R, y_R)$ lze pak odvodit jako

$$x_R = s^2 - x_P - x_Q,$$

$$y_R = s(x_P - x_R) - y_P.$$

- V případě $P = Q$ přechází jejich spojnice v tečnu ke křivce E a její směrnice je rovna $s = (3x_P^2 + a)/(2y_P)$. Souřadnice bodu $R = (x_R, y_R)$ získáme

$$x_R = \left(\frac{3x_P^2 + a}{2y_P}\right)^2 - 2x_P,$$

$$y_R = \left(\frac{3x_P^2 + a}{2y_P}\right)(x_P - x_R) - y_P.$$

7.3 Eliptické křivky nad $\mathbb{Z}_p$

Jak už jsme naznačili dříve, v kryptografii se používají eliptické křivky nad tělesem $GF(p) = \mathbb{Z}_p$ a křivky nad tělesem $GF(2^m)$. Nejprve se budeme věnovat tělesu $\mathbb{Z}_p$. Neexistuje vhodná geometrická interpretace pro operace na eliptické křivce konstruované nad konečným tělesem. Proto musíme operace interpretovat algebraicky, nicméně operace korespondují s operacemi pro eliptickou křivku nad reálnými čísly, viz kap. 7.2.

Eliptická křivka $E_p(a,b)$ nad tělesem $\mathbb{Z}_p$ je definována jako bod v nekonečnu $\mathcal{O}$ společně se všemi dvojicemi celých čísel (x,y), kde x a y jsou z tělesa $\mathbb{Z}_p$ a splňují rovnici $y^2 = x^3 + ax + b$ v F_p, tj. $y^2 \equiv x^3 + ax + b \pmod p$. Víme, že koeficienty a, b jsou také prvky tělesa $\mathbb{Z}_p$ a musí splňovat podmínku $4a^3 + 27b^2 \not\equiv 0 \pmod p$, která zaručuje, že takto definovaná množina bodů tvoří grupu (jinak koeficienty a a b můžeme volit libovolně (budou to veřejné parametry příslušného kryptosystému).

- V této grupě definujeme opačný bod k $\mathcal{O}$ jako $-\mathcal{O} = \mathcal{O}$ a pro ostatní nenulové $P = (x_P, y_P) \in E_p(a,b)$ definujeme $-P = (x_P, -y_P \pmod p)$, dále pro všechny body $P \in E_p(a,b)$ definujeme $P + -P = \mathcal{O}$ a $P + \mathcal{O} = P$.

- Sčítání různých nenulových a ne vzájemně opačných bodů $P = (x_P, y_P)$ a $Q = (x_Q, y_Q)$: jestliže P a Q jsou různé body takové, že $P \neq -Q$, pak $P+Q = R$, kde
$$s = (y_P - y_Q)/(x_P - x_Q) \pmod p,$$
$$x_R = s^2 - x_P - x_Q \pmod p,$$
$$y_R = -y_P + s(x_P - x_R) \pmod p.$$
Operaci dělení definujeme jako násobení inverzním prvkem, například x/y je $x \cdot y^{-1}$ a přirozeně y^{-1} je ten prvek tělesa $\mathbb{Z}_p$, pro který $y \cdot y^{-1} \equiv 1 \pmod p$.

- Zdvojení bodu P: pokud $y_P \neq 0$, pak $2P = R$, kde
$$s = (3x_P^2 + a)/(2y_P) \pmod p,$$
$$x_R = s^2 - 2x_P \pmod p,$$
$$y_R = -y_P + s(x_P - x_R) \pmod p.$$

Příklad 7.4. $(E_{23}(1,4))$. Nechť $p = 23$ a předpokládejme křivku $E : y^2 = x^3 + x + 4$ nad F_{23}, tj. $a = 1, b = 4$. Platí $4a^3 + 27b^2 = 4 + 432 = 436 \equiv 22 \pmod{23}$.

1. Body (viz obr. 7.6) této křivky jsou $\mathcal{O}$ a

$(0,2)$	$(0,21)$	$(1,11)$	$(1,12)$	$(4,7)$	$(4,16)$	$(7,3)$
$(7,20)$	$(8,8)$	$(8,15)$	$(9,11)$	$(9,12)$	$(10,5)$	$(10,18)$
$(11,9)$	$(11,14)$	$(13,11)$	$(13,12)$	$(14,5)$	$(14,18)$	$(15,6)$
$(15,17)$	$(17,9)$	$(17,14)$	$(18,9)$	$(18,14)$	$(22,5)$	$(22,18)$

2. Nechť $P = (4,7)$ a $Q = (13,11)$, potom $P + Q = R$ se vypočte takto:
$$x_R = ((11-7)/(13-4))^2 - 4 - 13 = 3^2 - 4 - 13 = -8 \equiv 15 \pmod{23},$$
$$y_3 = 3(4 - 15) - 7 = -40 \equiv 6 \pmod{23}.$$
Tedy $P + Q = (15,6)$.

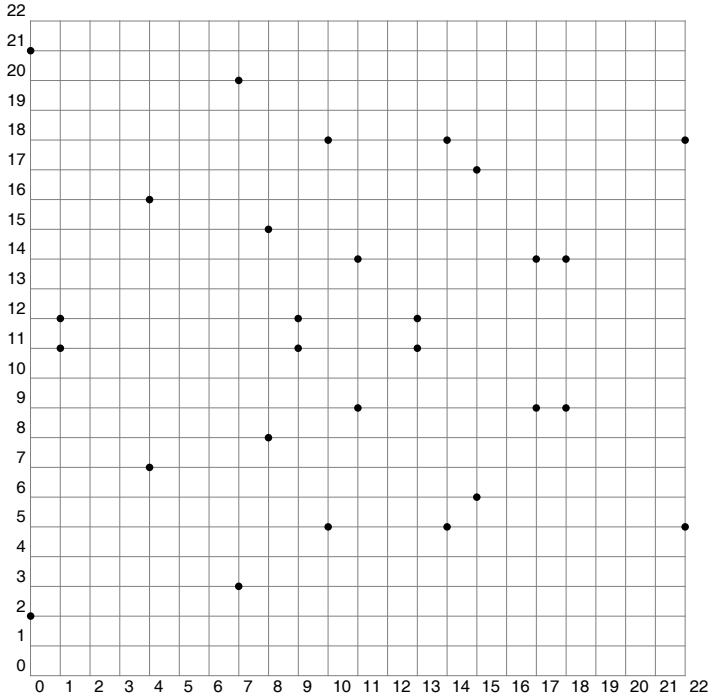
3. Nechť $P = (4, 7)$ potom $2P = P + P = R$ se vypočte takto:

$$x_R = ((34^2 + 1)/14)^2 - 8 = 15^2 - 8 = 217 \equiv 10 \pmod{23},$$

$$y_3 = 15(4 - 10) - 7 = -97 \equiv 18 \pmod{23}.$$

Tedy $2P = (10, 18)$.

▲



Obr. 7.6 Body křivky $E : y^2 = x^3 + x + 4$ nad F_{23}

Podobně jako $2P$ vypočítáme $3P = (P + P) + P = 2P + P, 4P, 5P \ldots$ Získáme obecně různé body xP na křivce $E_p(a, b)$. Protože má křivka konečný počet bodů, musí se po určitém počtu (l) kroků tato posloupnost zacyklit. V bodě zacyklení (lP) tak platí $lP = mP$, kde mP je nějaký dřívější bod. Odtud ale dostáváme

$$lP - mP = (l - m)P = \mathcal{O},$$

čili existuje nějaké $n = l - m, n < l$ takové, že $nP = \mathcal{O}$. Je tedy jasné, že v posloupnosti $P, 2P, 3P, 4P \ldots$ se vždy nakonec dostaneme do bodu $\mathcal{O}$ a poté cyklus začíná znovu od bodu P, neboť

$$(n + 1)P = nP + P = \mathcal{O} + P = P.$$

Definice 7.5. Nechť n je přirozené číslo, bod $P \in E$. Nejmenší takové n, pro něž je $nP = \mathcal{O}$, nazýváme *řád bodu* P.

Definice 7.6. *Řádem eliptické křivky* E rozumíme celkový počet bodů této křivky. Ozn. jej jako $\#E$.

Různé body na křivce E mohou mít různý řád. V kryptografické praxi vybíráme takový bod, jehož řád je roven největšímu prvočíslu v rozkladu čísla $\#E$ (pokud je n velké, například řádově 2^{256}, dostaneme velmi dlouhou posloupnost, než se „zacyklí“) nebo jeho násobku (tzv. kofaktoru, kofaktor $h = \#E/n$).

Příklad 7.7. ($E_{23}(1,4)$). Mějme opět křivku z předchozího příkladu($\mathbb{Z}_p$, $p = 23$, $E : y^2 = x^3 + x + 4$.) Řád křivky je $\#E = 29$. Mějme bod $P = (0, 2)$, jaký je jeho řád?

$1P = (0,2)$	$2P = (13,12)$	$3P = (11,9)$	$4P = (1,12)$
$5P = (7,20)$	$6P = (9,11)$	$7P = (15,6)$	$8P = (14,5)$
$9P = (4,7)$	$10P = (22,5)$	$11P = (10,5)$	$12P = (17,9)$
$13P = (8,15)$	$14P = (18,9)$	$15P = (18,14)$	$16P = (8,8)$
$17P = (17,14)$	$18P = (10,18)$	$19P = (22,18)$	$20P = (4,16)$
$21P = (14,18)$	$22P = (15,17)$	$23P = (9,12)$	$24P = (7,3)$
$25P = (1,11)$	$26P = (11,14)$	$27P = (13,11)$	$28P = (0,21)$
$29P = \mathcal{O}$			

▲

7.4 Eliptické křivky nad $GF(2^m)$

Pokud bychom chtěli pracovat s eliptickou křivkou E nad $GF(2^m)$, splňuje bod eliptické křivky $(x, y) \in E \setminus \{\mathcal{O}\}$ rovnici

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

nad $GF(2^m)$ a proměnné x, y a koeficienty a, b jsou prvky z $GF(2^m)$. To znamená, že všechny operace jsou prováděny pomocí polynomiální aritmetiky v $GF(2^m)$, viz kapitola 3.3.2.

Eliptická křivka $E_{2^m}(a, b)$ nad tělesem $GF(2^m)$ (bod v nekonečnu $\mathcal{O}$ společně se všemi dvojicemi celých čísel (x, y) splňující uvedenou rovnici) formuje opět Abelovskou grupu na množině $E_{2^m}(a, b)$, kde $b \neq 0$. Pro $\forall P, Q \in E_{2^m}(a, b)$ platí:

- $P + \mathcal{O} = P$.
- Nechť $P = (x_P, y_P)$, $-P = (x_P, x_P + y_P)$, pak $P + (-P) = \mathcal{O}$.

- Nechť $P = (x_P, y_P)$ a $Q = (x_Q, y_Q)$. Jestliže $P \neq Q$ a $P \neq -Q$, pak $P + Q = R = (x_R, y_R)$ se určí jako

$$s = (y_P + y_Q)/(x_P + x_Q),$$

$$x_R = s^2 + s + x_P + x_Q + a,$$

$$y_R = y_P + x_R + s(x_P + x_R).$$

- Nechť $P = (x_P, y_P)$, pak se $2P = R = (x_R, y_R)$ pro $x_P \neq 0$ určí jako

$$s = x_P + \frac{y_P}{x_P},$$

$$x_R = s^2 + s + a,$$

$$y_R = x_P^2 + (s+1)x_R.$$

7.5 Diskrétní logaritmus nad eliptickou křivkou

Pro kryptografické algoritmy na bázi eliptických křivek musíme najít výpočetně obtížný problém, který by korespondoval s problémem faktorizace velkého celého čísla (algoritmus RSA) nebo s problémem diskrétního logaritmu (např. algoritmus Diffie-Hellman). Mějme body $Q, R \in E_p(a, b)$ a rovnici $Q = kR$, kde $k < p$. Ze znalosti R a k snadno vypočítáme. Ale vypočítat k pouze ze znalosti Q a R je výpočetně obtížné. Kladné celé číslo k se nazývá *diskrétní logaritmus bodu Q vzhledem k základu R nad nad eliptickou křivkou E*. Uvedený problém se nazývá *problémem diskrétního logaritmu nad eliptickou křivkou* (ECDLP, eliptic curve discrete logarithm problem) a je analogií problému diskrétního logaritmu. Schéma diskrétního logaritmu (multiplikativní grupa) se transformuje na eliptickou křivku (aditivní grupa) tak, že operace násobení prvků $g \cdot g \cdot g \cdot \ldots$ (tj. g^k) se převede na sčítání bodů na křivce $P + P + P + \ldots$. (tj. kP).

Příklad 7.8. Mějme křivku $E_{23}(9, 17)$ s rovnicí $y^2 \pmod{23} = (x^3 + 9x + 17) \pmod{23}$. Jaký je diskrétní logaritmus k bodu $Q = (4, 5)$ vzhledem k základu $P = (16, 5)$. Můžeme použít hrubou sílu a počítat všechny násobky bodu P dokud nenalezneme bod Q. Tedy

$$P = (16, 5), 2P = (20, 20), 3P = (14, 14), 4P = (19, 20), 5P = (13, 10),$$

$$6P = (7, 3), 7P = (8, 7), 8P = (12, 17), 9P = (4, 5).$$

Protože $9P = (4, 5) = Q$, je diskrétní logaritmus k bodu $Q = (4, 5)$ vzhledem k základu $P = (16, 5) = 9$. Ve skutečnosti pracujeme s mnohem početnějšími tělesy, proto je útok hrubou silou nepoužitelný.
▲

Jednou z nejúčinnějších metod řešení úlohy ECDLP je Pollardova rho metoda, jejíž složitost je řádově $(\pi \cdot n/2)^{1/2}$ kroků. Pokud je $n = p = 2^{256}$ pro F_p, potřebujeme cca 2^{128} kroků, což je zhruba na úrovni luštitelnosti symetrické blokové šifry (viz tabulka 7.1) s 128-bitovým klíčem a z výpočetního hlediska neřešitelné. Proto říkáme, že příslušná šifra je výpočetně bezpečná. Úlohu lze sice paralelizovat, takže pokud použijeme N procesorů, dostáváme složitost $(\pi \cdot n/2)^{1/2}/N$, ale pro velká n je to stále výpočetně neřešitelná úloha.

Jen pro zajímavost uveďme, která tělesa doporučuje pro kryptosystémy americký NIST:

1. tělesa F_p pro $p = 2^{192} - 2^{64} - 1$, $p = 2^{224} - 2^{96} + 1$, $p = 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$, $p = 2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$, $p = 2^{521} - 1$,

2. tělesa $GF(2^m)$: $F_{2^{163}}$, $F_{2^{233}}$, $F_{2^{283}}$, $F_{2^{409}}$, $F_{2^{571}}$.

Příklady k procvičení

1. Nechť $E : y^2 = x^3 + 17$ je eliptická křivka nad $\mathbb{R}$ a nechť $P = (-2, 3)$, $Q = (1/4, 33/8)$ jsou dva body na této křivce. Najděte bod $R = P + Q$.

2. Nechť $E : y^2 = x^3 - 36x$ je eliptická křivka nad $\mathbb{R}$ a nechť $P = (-3, 9)$, $Q = (-2, 8)$ jsou dva body na této křivce. Najděte bod $R = P + Q$ a bod $2P$.

3. Nechť $E : y^2 = x^3 - 2x + 3$ je eliptická křivka nad $\mathbb{Z}_7$ a nechť $P = (1, 3)$ je bod na této křivce. Najděte bod $10P$.

4. Je dána křivka $E_{23}(1, 1)$, tj. $E : y^2 = x^3 + x + 1$, $\mathbb{Z}_p$, $p = 23$. Nalezněte všechny body této křivky. Určete řád této křivky. Zvolte si jeden z bodů této křivky a určete jeho řád.

5. Nechť $E : y^2 = x^3 + 3x$ je eliptická křivka nad $\mathbb{Z}_5$. Kolik má bodů a které to jsou? Tip: Začněte vyhodnocením pravé strany rovnice pro všechna x.

6. Nechť $E : y^2 = 3x^3 + 2x$ je eliptická křivka nad $\mathbb{Z}_5$. Kolik má bodů a které to jsou?

7. Nechť $E : y^2 = x^3 + 2x + 1$ je eliptická křivka nad $\mathbb{Z}_{13}$. Kolik má bodů a které to jsou?

8. Nechť $E : y^2 = x^3 + 4x$ je eliptická křivka nad $\mathbb{Z}_{13}$. Kolik má bodů a které to jsou?

Kapitola 8

Výsledky cvičení

Výsledky některých cvičení z kapitoly 1.2

ad 6. $1575 = 3^2 \cdot 5^2 \cdot 7^1$,
$23100 = 2^2 \cdot 3^1 \cdot 5^2 \cdot 7^1 \cdot 11^1$,
$\mathrm{NSD}(1575, 23100) = 2^0 \cdot 3^1 \cdot 5^2 \cdot 7^1 \cdot 11^0 = 525$
$\mathrm{NSN}(1575, 23100) = 2^2 \cdot 3^2 \cdot 5^2 \cdot 7^1 \cdot 11^1 = 69300$.

ad 7. $\phi(35) = \phi(7 \cdot 5) = \phi(7) \cdot \phi(5) = 6 \cdot 4 = 24$,
$\phi(64) = \phi(2^6) = (2-1) \cdot 2^5 = 32$,
$\phi(123) = \phi(3 \cdot 41) = \phi(3) \cdot \phi(41) = 2 \cdot 40 = 80$,
$\phi(600) = \phi(2^3 \cdot 3 \cdot 5^2) = 600 \cdot (1-1/2) \cdot (1-1/3) \cdot (1-1/5) = 160$ a $\phi(13) = 12$.

ad 8a) Vyplývá z vlastnosti Eulerovy funkce: pokud $\mathrm{NSD}(m, n) = 1$, pak $\phi(m \cdot n) = \phi(m) \cdot \phi(n)$.

ad 8b,c) vyplývá z vlastnosti $n = {p_1}^{a_1} \cdot {p_2}^{a_2} \cdot \ldots \cdot {p_k}^{a_k}$,
$\phi(n) = n \cdot (1 - \frac{1}{p_1}) \cdot (1 - \frac{1}{p_2}) \cdot \ldots \cdot (1 - \frac{1}{p_k}) =$
$= {p_1}^{a_1} \cdot {p_2}^{a_2} \cdot \ldots \cdot {p_k}^{a_k} \cdot (1 - \frac{1}{p_1}) \cdot (1 - \frac{1}{p_2}) \cdot \ldots \cdot (1 - \frac{1}{p_k}) =$
$= {p_1}^{a_1} \cdot (\frac{p_1 - 1}{p_1}) \cdot {p_2}^{a_2} \cdot (\frac{p_2 - 1}{p_2}) \cdot \ldots \cdot {p_k}^{a_k} \cdot (\frac{p_k - 1}{p_k})$.
Jestliže jediným prvočíselným dělitelem čísla n je $p_1 = 2$, pak $\phi(n) = 2^{a_1 - 1}(2 - 1)$, což je číslo sudé kvůli $2^{a_1 - 1}$. Jestliže máme prvočísla různá od 2, pak ${p_1}^{a_1} \cdot (p_1 - 1) \cdot {p_2}^{a_2} \cdot (p_2 - 1) \cdot \ldots \cdot {p_k}^{a_k} \cdot (p_k - 1)$ je sudé, protože jedno z prvočísel musí být liché a odečtením 1 dostaneme sudé číslo. V obou případech máme požadovaný výsledek.

ad 8d) Vyplývá z 8b) a 8c).

Výsledky některých cvičení z kapitoly 1.3

ad 1. Zbytkové třídy modulo 11 označme jako $[0]_{11}, [1]_{11}, \ldots, [10]_{11}$. Např. pak $[3]_{11} = \{\ldots, -18, -7, 3, 14, 25, \ldots\}$.

ad 2.,3.

a	0	1	2	3	4	5	6	7
$(-a)$	7	6	5	4	3	2	1	0
a^{-1}	–	1	–	3	–	5	–	7

ad 4.

a	0	1	2	3	4	5	6	7	8	9	10
a^{-1}	–	1	6	4	3	9	2	8	7	5	10

ad 5. Spočtěte $11^{13} \pmod{53}$. $11^2 = 121 \equiv 15 \pmod{53}$),
$11^4 = (11^2)^2 \equiv 15^2 = 225 \equiv 13 \pmod{53}$,
$11^8 = (11^4)^2 \equiv 13^2 = 169 \equiv 10 \pmod{53}$,
$11^{13} = 11^8 \cdot 11^4 \cdot 11^1 = (10 \cdot 13 \cdot 11) \equiv 52 \pmod{53}$,
Tj. $11^{13} \pmod{53} = 52$.

ad 6. Spočtěte $5^9 \pmod{42}$. $5^2 = 25 \pmod{42}$,
$5^4 = (5^2)^2 \equiv 25^2 = 625 \equiv 37 \pmod{42}$,
$5^8 = (5^4)^2 \equiv 37^2 = 1369 \equiv 25 \pmod{42}$,
$5^9 = 5^8 \cdot 5^1 = (25 \cdot 5) = 125 \equiv 41 \pmod{42}$,
Tj. $5^9 \pmod{42} = 41$.

Výsledky některých cvičení z kapitoly 2.2

ad 1a) Rozhodněte zda platí $3n = O(n)$. Ano, platí.

ad 1b) Rozhodněte zda platí $n^2 = O(n)$. Ne, neplatí.

ad 2. $2^{n+1} = O(2^n)$, ale $2^{2n} \neq O(2^n)$.
Abychom dokázali, že $2^{n+1} = O(2^n)$, musíme nalézt konstanty $c, n_0 > 0$ takové, že $0 \leqq 2^{n+1} \leqq c \cdot 2^n$ pro všechna $n \geqq n_0$. Protože $2^{n+1} = 2 \cdot 2^n$ pro všechna n, vyhovíme definici s $c = 2$ a $n_0 = 1$.
Abychom ukázali, že $2^{2n} \neq O(2^n)$, předpokládejme, že existují konstanty c, n_0 takové, že $0 \leqq 2^{2n} \leqq c \cdot 2^n$ pro všechna $n \geqq n_0$. Potom $2^{2n} = 2^n \cdot 2^n \leqq$ $\leqq c \cdot 2^n \Rightarrow 2^n \leqq c$. Ale žádná konstanta $c > 0$ není větší než všechna 2^n, a proto předpoklad vede ke sporu.

ad 7. $C_N \doteq (N+1) \cdot (N-1) \cdot log_3(N)$, protože vnější cyklus pro j proběhne $N+1$ krát, druhý cyklus pro k proběhne $N-1$ krát, a vnitřní cyklus proběhne cca $log_3(N)$ krát. Algoritmus je řádu $O(N^2 log(N))$.

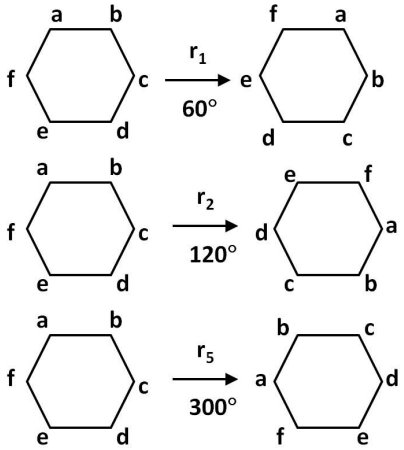
ad 9a) Předpokládejme, že se počet operací pro dvojnásobnou velikost vstupních dat zvětší o 1.
$C_1 = 0, C_N = C_{N/2} + 1$ pro $N > 1$.
$C_1 = 0$,
$C_2 = 1 + C_1 = 1 + 0 = 1$,
$C_4 = 1 + C_2 = 1 + 1 = 2$,
$C_8 = 1 + C_4 = 1 + 2 = 3, \ldots$

$N = 2^p : C_N = C_{2^p} = C_{(2^p)/2} + 1 = C_{2^{p-1}} + 1 = C_{2^{p-2}} + 2 = \ldots = C_{2^{p-(p-1)}} +$
$+ (p-1) = C_{2^1} + (p-1) = C_{2^0} + p = C_1 + p = p = log_2(N).$
Složitost algoritmu je $O(log_2(N))$.

ad 9b) Předpokládejme, že se počet operací pro dvojnásobnou velikost vstupních dat zvětší o N.
$C_1 = 0, C_N = C_{N/2} + N$ pro $N > 1$.
$C_1 = 0,$
$C_2 = 2 + C_1 = 2,$
$C_4 = 4 + C_2 = 4 + 2 = 6,$
$C_8 = 8 + C_4 = 8 + 4 + 2 = 14,$
$\ldots$
$C_N = N + \frac{N}{2} + \frac{N}{4} + \frac{N}{8} + \ldots + 2 = N * (1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \ldots) \doteq 2N.$
Složitost algoritmu je $O(N)$.

Výsledky některých cvičení z kapitoly 3.1

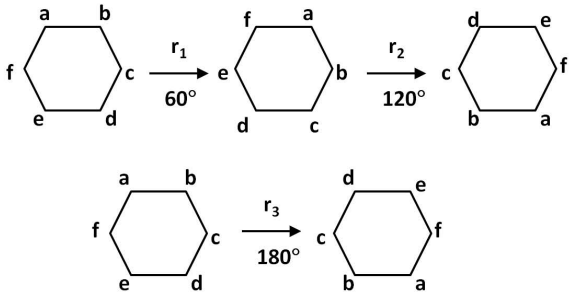
ad 4. Návod: ověřte zda množina $R = \{r_0, r_1, r_2, r_3, r_4, r_5\}$ (viz obr. 8.1) spolu s operací $r_i \circ r_j$ (viz obr. 8.2) splňuje požadované axiomy.



Obr. 8.1 Rotace šestiúhelníku r_1, r_2, r_5

ad 5. $\mathbb{Z}_{11}^*$ je cyklická, protože pro prvočíslo p je multiplikativni grupa $\mathbb{Z}_p^*$ cyklická, její řád je $\phi(11) = 10$. Generátor je např. $a = 2$, jeho řád je 10, protože $2^{\phi(11)} \pmod{11} = 1024 \pmod{11} = 1$, a $\phi(11)$ je nejmenší takový exponent, že obdržíme jako zbytek 1. Podgrupy a generátory viz tab. 8.1.

ad 6. $\mathbb{Z}_{10}^* = \{1, 3, 7, 9\}$ je cyklická, její řád je 4. Generátor je např. $a = 3$, jeho řád je $\phi(10) = 4$, což je nejmenší takový exponent, že obdržíme 1. Podgrupy a generátory viz tab. 8.2.



Obr. 8.2 Operace $\circ$ (skládání rotace): $r_1 \circ r_2$

podgrupa	řád	generátor
$\{1\}$	1	1
$\{1, 10\}$	2	10
$\{1, 3, 4, 5, 9, 10\}$	5	$3, 4, 5, 9$
$\{1, 2, \ldots, 10\}$	10	$2, 6, 7, 8$

Tab. 8.1 Podgrupy grupy $\mathbb{Z}_{11}^*$

podgrupa	řád	generátor
$\{1\}$	1	1
$\{1, 9\}$	2	9
$\{1, 3, 7, 9\}$	4	$3, 7, 9$

Tab. 8.2 Podgrupy grupy $\mathbb{Z}_{10}^*$

Výsledky některých cvičení z kapitoly 3.3

ad 1. $GF(2^4)$ má celkem 16 prvků, jsou to polynomy:
$0, 1, x, x^2, x^3, x+1, x^2+1, x^3+1, x^2+x, x^3+x, x^3+x^2, x^2+x+1,$
$x^3+x+1, x^3+x^2+x, x^3+x^2+1, x^3+x^2+x+1$

ad 2. Spočtěte obvyklým způsobem součet a součin následujících polynomů:
$(x^3+x^2+x)+(x^3+x+1) = (2x^3+x^2+2x+1)$
$(x^3+x^2+x)\cdot(x^3+x+1) = (x^6+x^5+2x^4+2x^3+2x^2+x)$

ad 3. Vyřešte v $GF(2^4)$ modulo ireducibilní polynom $m(x) = x^4+x+1$:
$(x^3+x^2+x)+(x^3+x+1) = (x^2+1)$,
bitově: $(1110) \oplus (1011) = (0101)$

ad 4. Vyřešte v $GF(2^4)$ modulo ireducibilní polynom $m(x) = x^4+x+1$:
$(x^3+x^2+x)\cdot(x^3+x+1) = (x^6+x^5+x) \pmod{(x^4+x+1)} = x^3$
bitově:
$1110 < 3 \oplus 1110 < 1 \pmod{1110} = 1110000 \oplus 11100 \oplus 1110 = 1100010$
$1100010 \pmod{10011} = 1100010 \oplus 10011 < 2 = 1100010 \oplus 1001100 = 101110$
$\pmod{10011} = 101110 \oplus 10011 < 1 = 101110 \oplus 100110 = 1000$

Výsledky některých cvičení z kapitoly 4.2

ad 1. Pomocí EA nalezněte NSD(12075, 4655).

a_0	$a_0 = 12075$
a_1	$a_1 = 4655$
$a_2 = a_0 \pmod{a_1}$	$a_2 = 12075 \pmod{4655} = 2765$
$a_3 = a_1 \pmod{a_2}$	$a_3 = 4655 \pmod{2765} = 1890$
$a_4 = a_2 \pmod{a_3}$	$a_4 = 2765 \pmod{1890} = 875$
$a_5 = a_3 \pmod{a_4}$	$a_5 = 1890 \pmod{875} = 140$
$a_6 = a_4 \pmod{a_5}$	$a_6 = 875 \pmod{140} = \mathbf{35} = \text{NSD}(12075, 4655)$
$a_7 = a_5 \pmod{a_6}$	$a_7 = 140 \pmod{35} = 0$

ad 2. Pomocí EEA nalezněte multiplikativní inverzní prvek k 299 (mod 323).

a_0	$a_0 = 323$
a_1	$a_1 = 299$
$a_2 = a_0 - q_1 \cdot a_1$	$a_2 = 323 - 1 \cdot 299 = 24$
$a_3 = a_1 - q_2 \cdot a_2$	$a_3 = 299 - 12 \cdot 24 = 11$
$a_4 = a_2 - q_3 \cdot a_3$	$a_4 = 24 - 2 \cdot 11 = 2$
$a_5 = a_3 - q_4 \cdot a_4$	$a_5 = 11 - 5 \cdot 2 = \mathbf{1} = \text{NSD}(323, 299)$

Protože je $\text{NSD}(323, 299) = 1$, můžeme 299^{-1} pomocí EEA hledat:
$a_0 = 323$
$a_1 = 299$
$a_2 = 24 = 323 - 1 \cdot 299$
$a_3 = 11 = 299 - 12 \cdot 24 = 299 - 12 \cdot (323 - 299) = 13 \cdot 299 - 12 \cdot 323$
$a_4 = 2 = 24 - 2 \cdot 11 = 12 \cdot (323 - 299) - (13 \cdot 299 - 12 \cdot 323) = 24 \cdot 323 - 25 \cdot 299$
$a_5 = \text{NSD}(323, 299) = 1 = 11 - 5 \cdot 2 = (13 \cdot 299 - 12 \cdot 323) - 5 \cdot (24 \cdot 323 - 25 \cdot 299) = 148 \cdot 299 - 137 \cdot 323,$
$x = 148, y = -137.$
Multiplikativní inverzní prvek k 299 je 148 (mod 323). Musí platit $a \cdot a^{-1} \equiv 1 \pmod{n}$ pro $a \in \mathbb{Z}, a \neq 0$, a to platí, $299 \cdot 148 \equiv 1 \pmod{323}$.

Výsledky některých cvičení z kapitoly 4.3

ad 1. Vypočítejme $3^{33} \pmod{17}$ pomocí RSMA.
Bitová reprezentace čísla $33 = 1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 100001$.
Pro účely algoritmu potřebujeme reprezentaci od nejméně významného bitu (tedy od LSB (nebo little-endian)), takže je opět $33 = 100001$.

i	0	1	2	3	4	5
k_i	1	0	0	0	0	1
A	3	9	13	16	1	1
b	3	3	3	3	3	$\mathbf{3} = 3^{33} \pmod{17}$

Jinak: $3^{33} \pmod{17}$.
$3^2 \equiv 9 \pmod{17}$,
$3^4 = (3^2)^2 \equiv 9^2 \equiv 13 \pmod{17}$,
$3^8 = (3^4)^2 \equiv 13^2 \equiv 16 \pmod{17}$,
$3^{16} = (3^8)^2 \equiv 16^2 \equiv 1 \pmod{17}$,
$3^{32} = (3^{16})^2 \equiv 1^2 \equiv 1 \pmod{17}$,
$3^{33} = 3^{32} \cdot 3^1 \equiv (1 \cdot 3) \equiv 3 \pmod{17}$,
Tj. $3^{33} \pmod{17} = 3$

ad 2b) $(x^2+1)^{11} \pmod{x^5+x^2+1} = x^4+x^3+x^2+x$.

i	0	1	2	3
k_i	1	1	0	1
A	(x^2+1)	(x^4+1)	(x^3+x^2)	(x^4+x^3+x)
b	(x^2+1)	$(x^4+x^3+x^2+x+1)$	$(x^4+x^3+x^2+x+1)$	$x^4+x^3+x^2+x$

Výsledky některých cvičení z kapitoly 5

ad 3. Např. $Q_{11} = \{1,3,4,5,9\}$, protože $\mathbb{Z}_{11}^* = \{1,2,3,4,5,6,7,8,9,10\}$ a dále:

$1^2 \pmod{11} = 1$,
$2^2 \pmod{11} = 4$,
$3^2 \pmod{11} = 9$,
$4^2 \pmod{11} = 5$,
$5^2 \pmod{11} = 3$,
$6^2 \pmod{11} = 3$,
$7^2 \pmod{11} = 5$,
$8^2 \pmod{11} = 9$,
$9^2 \pmod{11} = 4$,
$10^2 \pmod{11} = 1$.

Odmocnina $a = 3 \in \mathbb{Z}_{11}^*$ je číslo 5 a 6. Odmocnina $a = 5 \in \mathbb{Z}_{11}^*$ je číslo 4 a 7.

Výsledky některých cvičení z kapitoly 6.2

1 Nalezněte $3^{201} \pmod{11}$, použijte Fermatovu větu.
Platí, že $3 \equiv 3 \pmod{11}$. Protože $\mathrm{NSD}(3,11) = 1$ dostaneme z MFV $3^{10} \equiv 1 \pmod{11}$.
Dále $201 \equiv 5 \pmod{10}$, tj. $201 = (20) \cdot 10 + 1$, proto

$$\begin{aligned} 3^{201} &\equiv 3^{(20)\cdot 10+1} \pmod{11} \\ &\equiv \left(3^{10}\right)^{20} \cdot 3^1 \pmod{11} \\ &\equiv 1^{20} \cdot 3^1 \pmod{11} \\ &\equiv 3^1 \pmod{11} \end{aligned}$$

a $3^1 = 3 \equiv 3 \pmod{11}$. Proto $3^{201} \equiv 3 \pmod{11}$.

2 Nalezněte $28^{1202} \pmod{13}$, použijte Fermatovu větu.
Platí, že $28 \equiv 2 \pmod{13}$. Protože $\mathrm{NSD}(2,13) = 1$ dostaneme z MFV $28^{12} \equiv 2$

(mod 13).
Dále $1202 \equiv 2 \pmod{12}$, tj. $1202 = (100) \cdot 12 + 2$, proto

$$\begin{aligned} 2^{1202} &\equiv 2^{(100)\cdot 12+2} \pmod{13} \\ &\equiv \left(2^{12}\right)^{100} \cdot 2^2 \pmod{13} \\ &\equiv 1^{100} \cdot 2^2 \pmod{13} \\ &\equiv 2^2 \pmod{13} \end{aligned}$$

a $2^2 = 4 \equiv 4 \pmod{13}$. Proto $28^{1202} \equiv 4 \pmod{13}$.

3 Nalezněte $3^{201} \pmod{11}$, použijte Eulerovu větu.
Vidíme, že $\mathrm{NSD}(3, 11) = 1$ a $3 \equiv 3 \pmod{11}$.
Dále $\phi(11) = 10$ a platí, že $201 \equiv 1 \pmod{10}$.
Z předchozího plyne $3^{201} \pmod{11} \equiv 3^1 \pmod{11} = 3$. Tj. $3^{201} \equiv 3 \pmod{11}$.

4 Nalezněte $28^{1202} \pmod{13}$, použijte Eulerovu větu.
Vidíme, že $\mathrm{NSD}(28, 13) = 1$ a $28 \equiv 2 \pmod{13}$.
Dále $\phi(13) = 12$ a platí, že $1202 \equiv 2 \pmod{12}$.
Z předchozího plyne $28^{1202} \pmod{13} \equiv 2^2 \pmod{13} = 4$. Tj. $28^{1202} \equiv 4 \pmod{13}$.

5 Pomocí Eulerovy věty najděte multipikativní inverzní prvek k 4 modulo 11.
Je-li n malé, hledáme $1, \ldots, n-1$ dokud není nalezeno takové $a^{-1} \equiv a^{\phi(n)-1} \pmod{n}$. Takové a^{-1} pro náš příklad je 3, protože $\phi(11) = 10$ a $a^{-1} \equiv 4^{\phi(10)-1} \pmod{11} = 4^9 \pmod{11} = 3$. Ověříme, že platí $a \cdot a^{-1} \equiv 1 \pmod{n}$, tj. $4 \cdot 3 \equiv 1 \pmod{11}$.

6 Vyřešme kongruenci $3 \cdot x \equiv 4 \pmod{29}$. Protože $\mathrm{NSD}(3, 29) = 1$, má tato kongruence právě jedno řešení:

$$x \equiv 4 \cdot 3^{\phi(29)-1} \pmod{29} \equiv 4 \cdot 3^{27} \pmod{29} = 11.$$

Výsledky některých cvičení z kapitoly 6.3

ad 1. Řešením je $x \equiv 2 \cdot 35 \cdot 2 + 3 \cdot 21 \cdot 1 + 2 \cdot 15 \cdot 1 \pmod{105} \equiv 233 \pmod{105} \equiv 23 \pmod{105}$. Nejmenší celé číslo, které je řešením soustavy je 23.

ad 2. Řešením je $x \equiv 2 \cdot 36 \cdot 1 + 7 \cdot 28 \cdot 1 + 3 \cdot 63 \cdot 3 \pmod{252} \equiv 835 \pmod{252} \equiv 79 \pmod{252}$. Nejmenší celé číslo, které je řešením soustavy je 79.

ad 3. Řešením je $x \equiv 5 \cdot 143 \cdot 5 + 3 \cdot 91 \cdot 4 + 10 \cdot 77 \cdot 12 \pmod{1001} \equiv 13907 \pmod{1001} \equiv 894 \pmod{1001}$. Nejmenší celé číslo, které je řešením soustavy je 894.

ad 4. Řešení rozdělte do dvou kroků. Pondělí očíslujte 1, úterý 2, atd. Nejprve nalezněte den, kdy mají všichni profesoři přednášky (současně). Pak musíte zjistit, kdy tento den připadne na neděli. Řešení je $x = 371$.

ad 5. Vajec bylo 53. Je to řešení soustavy kongruencí
$x \equiv 1 \pmod 2$,
$x \equiv 2 \pmod 3$,
$x \equiv 3 \pmod 5$,
$x \equiv 4 \pmod 7$.

Výsledky některých cvičení z kapitoly 6.4

ad 1. Jaký je řád $3, 5, 7$ modulo 8?
Např. jaký je řád celého čísla $3 \pmod 8$?
$3^1 \equiv 3 \pmod 8$,
$3^2 = 9 \equiv 1 \pmod 8$,
$3^3 = 27 \equiv 3 \pmod 8$,
$3^4 = 81 \equiv 1 \pmod 8$.
Nemá smysl dále pokračovat, neboť posloupnost je periodická a délka periody je nejmenší exponent m takový, že $3^m \equiv 1 \pmod 8$. Tedy řád prvku $3 \pmod 8$ je $m = 2$, tj. $ord_8(3) = 2$

ad 2. Ověřte, zda je či není 17 primitivní odmocnina modulo 45. Je potřeba určit, zda $ord_{45}(17) = \phi(45)$. Protože $\phi(45) = 24$, měli bychom určit zda je $ord_{45}(17) = 24$. $17 \equiv 17 \pmod{45}$,
$17^2 \equiv 19 \pmod{45}$,
$17^3 \equiv 8 \pmod{45}$,
$17^4 \equiv 1 \pmod{45}$, tedy $ord_{45}(17) = 4$. Proto 17 není primitivní odmocninou modulo 45.

ad 5. Jaké jsou primitivní odmocniny modulo 47 a kolik jich je? Protože 47 je prvočíslo, existuje $\phi(47-1) = \phi(45) = 22$ primitivních odmocnin modulo 47. Jsou to $5, 10, 11, 13, 15, 19, 20, 22, 23, 26, 29, 30, 31, 33, 35, 38, 39, 40, 41, 43, 44, 45$.

ad 6. Nalezněte všechny primitivní odmocniny modulo 25. Nechť $n = 25$, pak existuje $\phi(\phi(25)) = 8$ primitivních odmocnin modulo 25. Přesněji to jsou $2, 3, 8, 12, 13, 17, 22, 23$.

ad 7. Nalezněte primitivní mocniny a modulo 17 a diskrétní logaritmy modulo 17. Existuje $\phi(17-1) = \phi(16) = 8$ primitivných odmocnin, jsou to $3, 5, 6, 7, 10, 11, 12, 14$. Diskrétní logaritmy - postupujte jako v tab. 6.3.

ad 9. Nalezněte diskrétní logaritmy x (pozor, kongruence nemusí mít řešení, proč?) pro:

- $3 \equiv 2^x \pmod 5$: řešení $x = 3$, číslo 2 je primitivn9 odmocnina modulo 5, $ord_5(2) = 4$, $\phi(5) = 4$.
- $2 \equiv 4^x \pmod{13}$: nemá řešení.

Výsledky některých cvičení z kapitoly 7

ad 3. Nechť $E : y^2 = x^3 - 2x + 3$ je eliptická křivka nad $\mathbb{Z}_7$ a nechť $P = (1, 3)$ je bod na této křivce. Najděte bod $10P$.
Řešení: $10P = (6, 5)$.

ad 7. Nechť $E : y^2 = x^3 + 2x + 1$ je eliptická křivka nad $\mathbb{Z}_{13}$. Kolik má bodů a které to jsou? Řešení:
Body této křivky jsou $\mathcal{O}$ a $(0, 1)$ $(0, 12)$ $(1, 2)$ $(1, 11)$ $(2, 0)$ $(8, 3)$ $(8, 10)$.

Literatura

[1] Agrawal M., Kayal N. a Saxena N. Primes in P. *Ann. of Math.* 160:2 (2004) 781–793.

[2] Cormen, T. H., Leiserson, C. E., Rivest, R.L. a Stein, C. *Introduction to Algorithms*. The MIT Press, 2001. ISBN 0262032937.

[3] Hankerson D., Menezes A.J. a Vanstone S.A. *Guide to Elliptic Curve Cryptography*. Springer, 2004. ISBN 978-0387952734.

[4] Kim S. H. a Pomerance C.. The probability that a random probable prime is composite. *Math. Comp.*. 53 (1989) 721–741.

[5] Menezes A.J., van Oorschot P.C. a Vanstone S.A. *Handbook of Applied Cryptography*. CRC Press, 2001. ISBN: 0-8493-8523-7. http://cacr.math.uwaterloo.ca/hac/, [cit. 2.2.2011].

[6] O'Neill, M. E. The genuine sieve of eratosthenes. *J. Funct. Program.* vol. 19 (2009), 95–106, ISSN 0956–7968, http://portal.acm.org/citation.cfm?id=1520298.1520303, [cit. 2.2.2011].

[7] Schneier B. *Applied Cryptography*. J. Willey, 1996, New York. ISBN 0-471-12845-7.

[8] Stallings W. *Cryptography and Network Security*. Prentice Hall, 1999, New Jersey. ISBN 0-13-869017-0.

[9] The Prime pages. http://primes.utm.edu/, [cit. 2.2.2011].

[10] Yan S. Y. *Number Theory for Computing*. Springer, 2nd ed., 2002. ISBN 978-3-540-43072-8.

Rejstřík

algoritmus
  Euklidův, 35
  Gaussův, 57
  pro rychlé modulární umocňování, 39
  rozšířený Euklidův, 37
  Trial division, 42
  Wheel factorization, 43
aritmetika
  modulární, 8
  polynomiální, 28

bod
  eliptické křivky, 69
  opačný, 70
  v nekonečnu, 69

charakteristika tělesa, 68

diskrétní logaritmus, 62
  nad nad eliptickou křivkou, 77
doba běhu, 13
dolní celá část, 8
délka
  periody, 60
dělitel, 5
  společný, 5
  triviální, 5

Eratosthenovo síto, 41
Eulerova funkce, 6

generátor grupy, 22
grupa, 20
  abelovská, 20
  aditivní, 21
  cyklická, 22
  multiplikativní, 21

kanonický tvar, 5
kofaktor, 76
kongruence, 8
křivka
  eliptická, 69
  rovinná, 67

lhář
  Fermatův, 44
  silný, 48

množina
  zbytkových tříd, 9
modul, 8
monoid, 21

nejmenší společný násobek, 6
největší společný dělitel, 6

odmocnina, 46
  primitivní, 61
okruh, 25
  komutativní, 25
  s jednotkou, 25

permutace, 2
  identická, 4
  inverzní, 4
podgrupa, 21
podíl, 5
pologrupa, 20
polynom, 28
  irecudibilní, 30
problém
  diskrétního logaritmu, 62

diskrétního logaritmu nad eliptickou křivkou, 77
prvek
aditivní opačný, 9
jednotkový, 21
multiplikativní inverzní, 9
nulový, 21
prvočíslo, 5
Mersennovo, 50
pseudoprvočíslo, 44
silné, 48

reflexivita, 17

složitost, 12
asymptotická, 15
O-značení, 15
o-značení, 15
Ω-značení, 15
ω-značení, 15
paměťová, 12
polynomiální, 16
subexponciální, 16
Θ-značení, 15
časová, 12
dolní odhad, 14
horní odhad, 14
řádová, 15
svědek
složenosti Fermatův, 44
složenosti silný, 48
symetrie, 17
transponovaná, 17
systém
zbytků redukovaný, 54
zbytků úplný, 54
sčítání bodů, 74

test
Fermatův, 44
Lucas-Lehmerův, 50
Miller-Rabinův, 46
tranzitivita, 17
trichotomie, 17

těleso, 25
Galoisovo, 27
konečné, 27

velikost vstupu, 13
věta
Dirichletova, 41
Eulerova, 55
Lagrangeova, 22
malá Fermatova, 52
prvočíselná, 41
Čínská o zbytcích, 57

zbytek, 5
kvadratický, 46
po dělení, 8
zdvojení bodu, 74

číslo
složené, 5
Carmichaelovo, 45
Mersennovo, 50
nesoudělné, 6

řád
a modulo n, 60
bodu, 76
eliptické křivky, 76
grupy, 21
konečného tělesa, 27
prvku, 22