Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE

Milan Straka

Kryptografie založená na kvadratických tělesech

Katedra algebry

Vedoucí diplomové práce: RNDr. David Stanovský, Ph.D.

Studijní program: matematické metody informační bezpečnosti

2008

Rád bych poděkoval vedoucímu své práce RNDr. Davidu Stanovskému, Ph.D., za odborné vedení během mé práce.

Také bych velmi rád poděkoval Petru Sušilovi za korektury textu a své přítelkyni Janě Kravalové kromě korektur za její trpělivost a lásku.

V neposlední řadě patří můj dík opět mým rodičům, kteří mě ochotně a bez přestání podporují celý můj život.

Prohlašuji, že jsem svou diplomovou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce a s jejím zveřejňováním.

Milan Straka

V Praze dne ....

Obsah

1. Úvod . . . 5
Značení a předpokládané znalosti . . . 6
2. Základy algebry kvadratických těles . . . 7
Vnoření, stopy, normy a diskriminanty . . . 7
Okruhy celistvých čísel kvadratických těles a jejich báze . . . 8
3. Třídová grupa imaginárních kvadratických těles . . . 13
Reprezentace ideálů imaginárních těles a operace nad nimi . . . 14
Ideály imaginárních těles jako pozitivní kvadratické formy . . . 16
Třídová grupa . . . 18
4. Algoritmy pro práci s třídovou grupou . . . 20
Přímočaré algoritmy v třídové grupě . . . 20
Praktické vylepšení přímočarých algoritmů . . . 25
Asymptoticky nejrychlejší algoritmy v třídové grupě . . . 26
Rozšířený Eukleidův algoritmus jako redukce dané formy . . . 30
5. Kryptografická primitiva třídové grupy . . . 33
Řešení IQ-DLP variantou kvadratického síta . . . 34
Volba kryptograficky vhodného diskriminantu třídové grupy . . . 37
Generování náhodných prvků s rovnoměrným rozdělením . . . 38
6. Kryptografické protokoly v třídové grupě . . . 40
Protokoly pro výměnu klíče . . . 40
Protokoly asymetrického šifrování . . . 40
Protokoly pro digitální podpis . . . 41
Útok na protokol IQ-RDSA . . . 43
7. Implementace knihovny primitiv a protokolů . . . 49
8. Použitá literatura . . . 52

Název práce: Kryptografie založená na kvadratických tělesech

Autor: Milan Straka

Katedra (ústav): Katedra algebry

Vedoucí diplomové práce: RNDr. David Stanovský, Ph.D.

E-mail vedoucího: *David.Stanovsky@mff.cuni.cz*

Abstrakt: Imaginární kvadratická tělesa byla navržena pro použití v asymetrické kryptografii Buchmannem a Williamsem již v roce 1988 a od té doby vznikly i další kryptografické protokoly. I když tyto protokoly nejsou tak efektivní jako podobná schémata s eliptickými křivkami, mohou konkurovat schématům založeným na RSA, a navíc je jejich bezpečnost považována za nezávislou na bezpečnosti běžných kryptosystémů jako RSA, DSA a ECC.

Tato práce shrnuje dosavadní výsledky v oboru kvadratické kryptografie. Jednak popisuje algebraickou teorii nutnou pro zavedení třídové grupy imaginárních kvadratických těles a dále studuje algoritmy operací v třídové grupě, jak asymptoticky, tak prakticky efektivní. Také rozebírá vhodná kryptografická schémata a útoky na ně.

Součástí této práce je knihovna, která popsané protokoly efektivně implementuje.

Klíčová slova: třídová grupa imaginárního kvadratického tělesa, diskrétní logaritmus, asymetrická kryptografie, šifrovací a podpisové schéma

Title: Qudratic field based cryptography

Author: Milan Straka

Department: Department of Algebra

Supervisor: RNDr. David Stanovský, Ph.D.

Supervisor's e-mail address: *David.Stanovsky@mff.cuni.cz*

Abstract: Imaginary quadratic fields were first suggested as a setting for public-key cryptography by Buchmann and Williams already in 1988 and more cryptographic schemes followed. Although the resulting protocols are currently not as efficient as those based on elliptic curves, they are comparable to schemes based on RSA and, moreover, their security is believed to be independent of other widely-used protocols including RSA, DSA and elliptic curve cryptography.

This work gathers present results in the field of quadratic cryptography. It recapitulates the algebraic theory needed to work with the class group of imaginary quadratic fields. Then it investigates algorithms of class group operations, both asymptotically and practically effective. It also analyses feasible cryptographic schemes and attacks upon them.

A library implementing described cryptographic schemes is a part of this work.

Keywords: class group of imaginary quadratic field, discrete logarithm, public-key cryptography, encryption and signature scheme

1. Úvod

Asymetrická kryptografie, nebo také kryptografie s veřejným klíčem, hraje základní roli v elektronické komunikaci. Jednak umožňuje šifrovanou komunikaci bez potřeby sdíleného tajemství mezi komunikujícími stranami a jednak představuje základní kámen pro schémata digitálního podpisu.

Asi nejpoužívanější algoritmus asymetrické kryptografie, a to jak šifrovací, tak podpisový, je RSA. Tento algoritmus je založen na výpočetně náročném problému, na faktorizaci daného čísla – najít prvočísla p a q, pokud dostaneme součin $n = pq$. Tento problém je možné řešit triviálně v exponenciálním čase, ale existují i sofistikované algoritmy, které umožňují řešení v lepším než exponenciálním čase. Nejrychlejší známý algoritmus je číselně teoretické síto, jehož složitost můžeme velmi hrubě přirovnat k $\mathcal{O}(e^{c\sqrt[3]{\ln x}})$.

Bohužel není znám žádný dolní odhad na složitost problému faktorizace, takže teoreticky může existovat faktorizační algoritmus pracující v polynomiálním čase. Navíc bezpečnost algoritmu RSA není ekvivalentní s problémem faktorizace – stačí dokázat počítat e-té odmocniny modulo $n = pq$. Tedy bezpečnost algoritmu RSA není nijak teoreticky podložena. Z tohoto důvodu je třeba hledat jiné algoritmy a výpočetní problémy pro asymetrickou kryptografii, které by byly nezávislé na problému faktorizace.

Jiný vhodný výpočetně náročný problém je problém diskrétního logaritmu. Pokud máme multiplikativní grupu G a prvky $g \in G$ a $g^x \in G$, problém diskrétního logaritmu spočívá v nalezení čísla x.

Obtížnost tohoto problému vždy závisí na grupě, ve které je tento problém definován. Například v aditivní grupě $\mathbb{Z}$ či $\mathbb{Z}_n$ je tento problém triviálně řešitelný – stačí použít operaci dělení. Ovšem už v multiplikativní grupě $\mathbb{Z}_p$ pro p prvočíslo je tento problém výpočetně velmi náročný a je na něm postaven algoritmus DSA. Nejrychlejší známý algoritmus na řešení diskrétního logaritmu v $\mathbb{Z}_p$ je varianta číselně teoretického síta, která má stejnou složitost jako původní faktorizační algoritmus, přičemž složitost vztahujeme k velikosti grupy, ve které diskrétní logaritmus hledáme.

Existují však i jiné grupy, ve kterých se zdá být problém diskrétního logaritmu výpočetně náročný. Jedním z příkladů je třídová grupa eliptických křivek nad konečnými tělesy. Nejrychlejší známý způsob řešení problému diskrétního logaritmu v této grupě je triviální algoritmus se složitostí $\mathcal{O}(\sqrt{x})$. Další vhodná grupa je třídová grupa imaginárních kvadratických těles, kterou budeme zkoumat v této práci. Řešením problému diskrétního logaritmu v této grupě lze získat faktorizaci jejího diskriminantu, takže problém diskrétního logaritmu v této grupě je alespoň tak výpočetně náročný jako problém faktorizace. Nejrychlejší známý algoritmus řešení problému diskrétního logaritmu v třídové grupě je varianta kvadratického síta, jejíž složitost lze velmi hrubě vyjádřit jako $\mathcal{O}(e^{c\sqrt[2]{\ln x}})$.

Tato práce shrnuje dosavadní výsledky v oboru kryptografie založené na třídové grupě imaginárních kvadratických těles. I když navrhované protokoly nejsou v tuto chvíli výrazně výhodnější než protokoly založené na RSA či eliptické kryptografii, má jejich studium velký význam, protože se bezpečnost těchto protokolů považuje za nezávislou na bezpečnosti ostatních protokolů.

V následujících dvou kapitolách zavedeme algebraickou teorii, která bude definovat a popisovat vlastnosti třídové grupy imaginárního kvadratického tělesa. V další kapitole se naučíme efektivně provádět operace v třídové grupě. Následně popíšeme výpočetně náročné problémy třídové grupy a jejich nejefektivnější známá řešení. Na základě těchto problémů pak v další kapitole navrhneme vhodné kryptografické protokoly. Nakonec vše skloubíme dohromady a vytvoříme vlastní implementaci nejvhodnějších kryptografických protokolů.

Značení a předpokládané znalosti

V celé práci budeme předpokládat základní znalosti a zavedené značení z následujících oblastí: dělitelnost a jednoznačné rozklady, ideály, minimální polynomy, tělesová rozšíření, jednoduchá rozšíření a konstrukce kořenových nadtěles, $\mathbb{Z}$-moduly a volné abelovské grupy. V případě nejasnosti značení je možné konzultovat například učebnice [30] či [42].

Pro značení logaritmu používáme tři funkce – ln pro přirozený logaritmus, lg pro logaritmus o základu dva a log pro logaritmus s neurčeným základem.

2. Základy algebry kvadratických těles

V této kapitole stručně shrneme základní definice a věty týkající se kvadratických těles, celistvých bází a okruhů celistvých čísel. Podrobnosti je možné najít v mnoha učebnicích algebry, například v [30] nebo [42].

Definice 2.1: *Algebraickým číslem stupně d* nazveme takové $\alpha \in \mathbb{C}$, které je kořenem nějakého polynomu m_α nad $\mathbb{Q}$ stupně d a není kořenem žádného polynomu nad $\mathbb{Q}$ menšího stupně. Polynom m_α nazýváme *minimální polynom prvku α* a jeho kořeny jsou s prvkem α *konjugované (nad $\mathbb{Q}$)*. Nejmenší těleso, které obsahuje $\mathbb{Q}$ i α, $\mathbb{Q}(\alpha)$, nazývejme *číselným tělesem stupně d*. Speciálně pojem *kvadratickým tělesem* označuje číselné těleso stupně dva. Pokud je imaginární část α nulová, hovoříme o *reálném* (kvadratickém) tělese, v opačném případě o *imaginárním*.

Celistvým číslem stupně d nazveme takové $\alpha \in \mathbb{C}$, jehož minimální polynom je monický s koeficienty v $\mathbb{Z}$.

Tvrzení 2.2: Všechna kvadratická tělesa můžeme zapsat jako $\mathbb{Q}(\sqrt{D})$, kde $D \in \mathbb{Q}$ je bezčtvercové racionální číslo.

Důkaz: Máme-li $\mathbb{Q}(\alpha)$ s minimálním polynomem $ax^2 + bx + c$, tak je zřejmě $\mathbb{Q}(\alpha)$ isomorfní $\mathbb{Q}(\sqrt{b^2 - 4ac})$. Pokud navíc $D = a^2 D_0$, tak je $\mathbb{Q}(\sqrt{D})$ isomorfní $\mathbb{Q}(\sqrt{D_0})$. ⊠

Vnoření, stopy, normy a diskriminanty

Definice 2.3: *Vnoření* číselného tělesa F (do $\mathbb{C}$) označme každý okruhový homomorfismus $\theta : F \to \mathbb{C}$. Pokud je $K \leq F$ rozšíření číselných těles, θ vnoření F a $\theta|_K$ je identita, mluvíme o vnoření jako o *K-isomorfismu*. Z definice je každé vnoření θ vždy $\mathbb{Q}$-isomorfismus.

Tvrzení 2.4: Číselné těleso $\mathbb{Q}(\alpha)$ stupně d má právě d vnoření $\theta_1, \ldots, \theta_d$ a obrazy $\theta_1(\alpha), \ldots, \theta_j(\alpha)$ jsou právě všechny kořeny m_α.

Důkaz: Protože θ_i fixuje celé $\mathbb{Q}$, $\theta_i(m_\alpha) = m_\alpha$ a $\theta_i(\alpha)$ je také kořen m_α. Naopak, pokud definujeme

$$\theta_i(f(\alpha)) = f(\alpha_i) \text{ pro } f \in \mathbb{Q}[x] \text{ a } \alpha_i \text{ } i\text{-tý kořen } m_\alpha,$$

získáme zajisté dobře definované vnoření.

⊠

Definice 2.5: Vnoření nazveme *reálné*, pokud $\theta(F) \subseteq \mathbb{R}$, v opačném případě jde o vnoření *imaginární*. *Signaturou* číselného tělesa F myslíme $\{r_1, r_2\}$, kde r_1 je počet reálných a r_2 polovina počtu imaginárních vnoření (polovina proto, že konjugované imaginární vnoření je také imaginární vnoření), takže $r_1 + 2r_2 = [F : \mathbb{Q}]$. V případě kvadratických těles vidíme, že reálná tělesa mají dvě reálná vnoření a imaginární tělesa mají dvě navzájem konjugovaná komplexní vnoření.

Tvrzení 2.6: Je-li $F \leq E$ rozšíření číselných těles, pak se každé vnoření F rozšiřuje do $[E : F]$ vnoření E. Speciálně existuje $[E : F]$ F-isomorfismů E.

Důkaz: Dokazujme indukcí dle $[E : F]$. Je-li E isomorfní F, tvrzení zřejmě platí. Předpokládejme tedy, že tvrzení platí pro všechna rozšíření číselných těles s menším stupněm rozšíření než $[E : F]$. Zvolme $\alpha \in E \setminus F$ a zajímejme se o rozšíření

$\mathbb{Q} \leq F(\alpha) \leq E$. Počet vnoření $F(\alpha)$ je $[F(\alpha) : \mathbb{Q}]$ dle (2.4) a dle indukčního předpokladu se každé z nich rozšiřuje do $[E : F(\alpha)]$ vnoření E. Takto jsme získali $[E : \mathbb{Q}]$ vnoření, čili všechna vnoření. Zbývá určit F-isomorfismy E. Ty získáme tak, že rozšíříme F-isomorfismy $F(\alpha)$, kterých je $[F(\alpha) : F]$ (jednoduché rozšíření (2.4)), takže dostáváme

$$[E : F(\alpha)] \cdot [F(\alpha) : F] = [E : F]$$

F-isomorfismů E.

$\boxtimes$

Definice 2.7: Mějme F číselné těleso stupně d s vnořeními $\theta_1, \ldots, \theta_d$. Pak pro $a \in F$ definujeme *stopu* $T_F(a)$ a *normu* $N_F(a)$ jako

$$T_F(a) = \sum_{j=1}^{d} \theta_j(a),$$

$$N_F(a) = \prod_{j=1}^{d} \theta_j(a).$$

Pozorování 2.8: Mějme číselné těleso F stupně n a $\alpha \in F$ s $[\mathbb{Q}(\alpha) : \mathbb{Q}] = d$, kde $\alpha_1, \ldots, \alpha_d$ jsou všechny kořeny m_α. Pak z tvrzení (2.4) a (2.6) plyne, že

$$T_F(\alpha) = \frac{n}{d} T_{\mathbb{Q}(\alpha)}(\alpha) = \frac{n}{d} \sum_{j=1}^{d} \alpha_j,$$

$$N_F(\alpha) = (N_{\mathbb{Q}(\alpha)}(\alpha))^{n/d} = \left(\prod_{j=1}^{d} \alpha_j \right)^{n/d}.$$

Jelikož $\mathbb{Q}[x] \ni m_\alpha = x^d - T_{\mathbb{Q}(\alpha)}(\alpha) + \ldots \pm N_{\mathbb{Q}(\alpha)}(\alpha)$, je $T_F(\alpha) \in \mathbb{Q}$ i $N_F(\alpha) \in \mathbb{Q}$.

Definice 2.9: Pro polynom $f \in F[x]$ stupně d nad číselným tělesem F definujeme *diskriminant* jako

$$\operatorname{disc}(f) = a^{2d-2} \prod_{1 \leq i < j \leq d} (\alpha_i - \alpha_j)^2,$$

kde α_i jsou kořeny polynomu f v $\mathbb{C}$. Speciálně pro $f = ax^2 + bx + c$ je $\operatorname{disc}(f) = b^2 - 4ac$.

Definice 2.10: Je-li $B = \{\alpha_1, \ldots, \alpha_d\}$ báze číselného tělesa F jako vektorového prostoru nad $\mathbb{Q}$, tak *diskriminant* této báze je

$$\operatorname{disc}(B) = \det(A)^2,$$

kde $A = (\theta_j(\alpha_i))_{ij}$ je čtvercová matice o rozměrech $d \times d$ a θ_j jsou vnoření F.

Pozorování 2.11: Je-li $B = \{1, \alpha, \ldots, \alpha^{d-1}\}$ báze číselného tělesa $\mathbb{Q}(\alpha)$, pak

$$\operatorname{disc}(B) = \det\left(\theta_j(\alpha^{i-1})\right)^2 = \det\left(\alpha_j^{i-1}\right)^2 \stackrel{V}{=} \prod_{1 \leq i < j \leq d} (\alpha_i - \alpha_j)^2 = \operatorname{disc}(m_\alpha),$$

kde α_j jsou všechny kořeny monického polynomu m_α (použili jsme vzoreček pro determinant Vandermondovy matice).

Okruhy celistvých čísel kvadratických těles a jejich báze

Definice 2.12: Všechna celistvá čísla v $\overline{\mathbb{Q}}$ (algebraickém uzávěru $\mathbb{Q}$), tj. všechna algebraická čísla nad $\mathbb{Q}$ s celočíselným monickým minimálním polynomem, budeme značit $\mathbb{A}$.

Pozorování 2.13: Pro $\alpha \in \mathbb{A}$ je $T(\alpha), N(\alpha) \in \mathbb{Z}$, protože odpovídají koeficientům minimálního polynomu $m_\alpha \in \mathbb{Z}[x]$.

Tvrzení 2.14: $\mathbb{A}$ je podokruh $\overline{\mathbb{Q}}$ (algebraického uzávěru $\mathbb{Q}$).

Důkaz: Je třeba dokázat jenom to, že pro $\alpha, \beta \in \mathbb{A}$ je $\alpha+\beta \in \mathbb{A}$ i $\alpha\beta \in \mathbb{A}$. Protože ale jsou α a β celistvá čísla, jsou $\mathbb{Z}[\alpha]$ i $\mathbb{Z}[\beta]$ jako $\mathbb{Z}$-moduly konečně generované. Takže také $\mathbb{Z}[\alpha, \beta]$ je konečně generovaný $\mathbb{Z}$-modul, který obsahuje $\alpha + \beta$ i $\alpha\beta$. Uvažujme libovolný prvek $\gamma \in \mathbb{Z}[\alpha, \beta]$ a ukažme, že je celistvý . Nechť $m_1, \ldots, m_d$ jsou všechny generátory $\mathbb{Z}[\alpha, \beta]$. Každé $\gamma m_i \in \mathbb{Z}[\alpha, \beta]$, takže

$$\gamma m_i = \sum_{j=1}^{d} a_{i,j} m_j.$$

Máme tedy soustavu rovnic

$$\left(\sum_{j=1}^{d} a_{i,j} m_j \right) - \gamma m_i$$

s nulovou pravou stranou, která má netriviální řešení $m_1, \ldots, m_d$. Její determinant, který pochopíme jako polynom v proměnné γ, je monický, má stupeň d a je roven nule. Z Gaussova lemmatu tedy máme, že γ je celistvé číslo.

$\boxtimes$

Definice 2.15: Buď F číselné těleso. Průnik $F \cap \mathbb{A}$ označme $\mathbb{Z}_F$. Protože $\mathbb{A}$ je podokruh $\overline{\mathbb{Q}}$, je i $\mathbb{Z}_F$ okruh, který budeme nazývat *okruh celistvých čísel F*. V literatuře se často setkáváme se značením $\mathfrak{O}_F$. Evidentně je $\mathbb{Z}_Q = \mathbb{Z}$.

Pozorování 2.16: Je-li F číselné těleso, tak pro každé $\beta \in F$ existuje $z \in \mathbb{Z}$, že $z\beta \in \mathbb{Z}_F$. To proto, že označíme-li

$$m_\beta = x^d + a_{d-1}x^{d-1} + \ldots + a_1 x + a_0,$$

je minimální polynom $z\beta$ roven

$$m_{z\beta} = x^d + z a_{d-1} x^{d-1} + \ldots + z^{d-1} a_1 x + z^d a_0.$$

Nyní spějeme k definici celistvé báze. K tomu budeme potřebovat dvě tvrzení o diskriminantech bází.

Tvrzení 2.17: Je-li $B = \{\alpha_1, \ldots, \alpha_d\}$ báze číselného tělesa F jako $\mathbb{Q}$-vektorového prostoru, tak pro její diskriminant platí

$$\operatorname{disc}(B) = \det\left(T_F(\alpha_i\alpha_j)\right) \in \mathbb{Q} \setminus \{0\}.$$

Důkaz: Víme, že $\operatorname{disc}(B) = \det(\theta_j(\alpha_i))^2$, a protože součin determinantů je determinant součinu a protože determinant je invariantní vůči transpozici, dostaneme

$$\det\left(\theta_j(\alpha_i)\right)^2 = \det\left(\sum_{k=1}^{d} \theta_k(\alpha_i\alpha_j)\right) = \det\left(T_F(\alpha_i\alpha_j)\right).$$

Poslední matice má všechny koeficienty v $\mathbb{Q}$, tedy i její determinant je v $\mathbb{Q}$. Navíc pokud by matice $(\theta_j(\alpha_i))$ byla singulární, tak protože jedno vnoření je identické, dostali bychom netriviální lineární kombinaci α_i, což je spor, proto je determinant a tedy i diskriminant nenulový.

$\boxtimes$

Tvrzení 2.18: Jsou-li $B_1 = \{\alpha_1, \ldots, \alpha_d\}$ a $B_2 = \{\beta_1, \ldots, \beta_d\}$ dvě báze číselného tělesa F jako $\mathbb{Q}$-vektorového prostoru, pak

$$\mathrm{disc}(B_2) = D^2 \mathrm{disc}(B_1),$$

kde $D = \det(q_{k,i})$ pro $\beta_k = \sum_{i=1}^{d} q_{k,i}\alpha_i$, přičemž všechny $q_{k,i} \in \mathbb{Q}$.

Důkaz: Označme θ_j všechna vnoření F. Použijeme-li tato vnoření na rovnosti

$$\beta_k = \sum_{i=1}^{d} q_{k,i}\alpha_i,$$

získáme

$$\theta_j(\beta_k) = \sum_{i=1}^{d} q_{k,i}\theta_j(\alpha_i),$$

které můžeme přepsat do matic jako

$$\begin{pmatrix} \theta_1(\beta_1) & \theta_2(\beta_1) & \cdots & \theta_d(\beta_1) \\ \theta_1(\beta_2) & \theta_2(\beta_2) & \cdots & \theta_d(\beta_2) \\ \vdots & \vdots & \ddots & \vdots \\ \theta_1(\beta_d) & \theta_2(\beta_d) & \cdots & \theta_d(\beta_d) \end{pmatrix} = \begin{pmatrix} q_{1,1} & \cdots & q_{1,d} \\ q_{2,1} & \cdots & q_{2,d} \\ \vdots & \ddots & \vdots \\ q_{d,1} & \cdots & q_{d,d} \end{pmatrix} \begin{pmatrix} \theta_1(\alpha_1) & \cdots & \theta_d(\alpha_1) \\ \theta_1(\alpha_2) & \cdots & \theta_d(\alpha_2) \\ \vdots & \ddots & \vdots \\ \theta_1(\alpha_d) & \cdots & \theta_d(\alpha_d) \end{pmatrix}.$$

Nyní doplníme determinanty a umocníme na druhou, což dá $\mathrm{disc}(B_2) = D^2\mathrm{disc}(B_1)$.

$\boxtimes$

Definice 2.19: Buď F číselné těleso. *Celistvou bází* F myslíme bázi $\mathbb{Z}_F$ nad $\mathbb{Z}$. Z následujícího tvrzení a pozorování (2.16) plyne, že celistvá báze je báze F jako $\mathbb{Q}$-vektorového prostoru, takže označení *celistvá báze* F je smysluplné.

Tvrzení 2.20: Každé číselné těleso F stupně d má celistvou bázi a $\mathbb{Z}_F$ je volná abelovská grupa ranku d.

Důkaz: Díky pozorování (2.16) víme, že existují báze F skládající se výhradně z prvků $\mathbb{Z}_F$. Díky tvrzení (2.17) víme, že takové báze mají nenulový celočíselný diskriminant. Vyberme si tedy takovou bázi $B = \{\beta_1, \ldots, \beta_d\} \subseteq \mathbb{Z}_F$, jejíž $|\mathrm{disc}(B)|$ je minimální. Sporem ukažme, že B je báze $\mathbb{Z}_F$.

Pokud není, existuje $\gamma \in \mathbb{Z}_F$, že

$$\gamma = \sum_{j=1}^{d} q_j\beta_h$$

a alespoň jedno $q_j \notin \mathbb{Z}$. Bez újmy na obecnosti nechť je to

$$q_1 = \lfloor q_1 \rfloor + r, \text{ kde } 0 < r < 1.$$

Uvažujme

$$\gamma - \lfloor q_1 \rfloor \beta_1 = \sum_{j=1}^{d} q_j\beta_j - \lfloor q_1 \rfloor \beta_1 = r\beta_1 + \sum_{j=2}^{d} q_j\beta_j.$$

Protože jak všechna β_i, tak γ i $\lfloor q_1 \rfloor$ náleží do $\mathbb{Z}_F$, také $r\beta_1 \in \mathbb{Z}_F$. Pak je ale $B' = \{r\beta_1, \beta_2, \ldots, \beta_d\}$ také báze, jejíž prvky náleží do $\mathbb{Z}_F$. Přitom ale dle (2.18) je $\mathrm{disc}(B') = r^2 \mathrm{disc}(B)$, což je spor s minimalitou diskriminantu B.

Tedy B je celistvá báze a $\mathbb{Z}_F$ se jako $\mathbb{Z}$-modul rovná

$$\mathbb{Z}_F = \mathbb{Z}\beta_1 \oplus \cdots \oplus \mathbb{Z}\beta_d$$

a je to tedy volná abelovská grupa ranku d.

$\boxtimes$

Je jednoduché si uvědomit, že pokud máme bázi F složenou z celistvých prvků, jejíž diskriminant je beze čtverců, tak je tato báze už celistvá. Opačná implikace ale neplatí.

Tvrzení 2.21: Buď B_1 a B_2 dvě celistvé báze číselného tělesa F. Pak

$$\mathrm{disc}(B_1) = \mathrm{disc}(B_2).$$

Důkaz: Z tvrzení (2.18) víme, že $\mathrm{disc}(B_2) = D^2 \mathrm{disc}(B_1)$. Protože jde ale o báze celistvé, tak jak diskriminanty, tak D jsou celá čísla. Proto $\mathrm{disc}(B_1) \mid \mathrm{disc}(B_2)$. Nyní stačí obrátit role B_1 a B_2 a dostaneme $\mathrm{disc}(B_2) \mid \mathrm{disc}(B_1)$. Protože jsou ale diskriminanty celočíselné, musí být $\mathrm{disc}(B_2) = \pm\mathrm{disc}(B_1)$. Tedy $D^2 = \pm 1$ a proto jsou si diskriminanty rovny.

$\boxtimes$

Předchozí dvě tvrzení nám umožnily následující definici:

Definice 2.22: *Diskriminant číselného tělesa F, Δ_F,* buď diskriminant libovolné celistvé báze F.

Nyní v případě kvadratických těles charakterizujeme jak okruh celistvých čísel, tak diskriminant kvadratického tělesa.

Tvrzení 2.23: Buď $D \neq 1$ bezčtvercové celé číslo a označme $F = \mathbb{Q}(\sqrt{D})$ kvadratické těleso s diskriminantem Δ_F. Pak

$$\Delta_F = \begin{cases} D & \text{když } D \equiv 1 \pmod 4 \\ 4D & \text{když } D \equiv 2,3 \pmod 4 \end{cases} \quad \text{a} \quad \mathbb{Z}_F = \begin{cases} \mathbb{Z}[\frac{1+\sqrt{D}}{2}] & \text{když } \Delta_F \equiv 1 \pmod 4 \\ \mathbb{Z}[\sqrt{D}] & \text{když } \Delta_F \equiv 0 \pmod 4 \end{cases}.$$

Důkaz: Buď $\mathbb{Z}_F = \mathbb{Z}[\alpha]$. Bez újmy na obecnosti můžeme jistě předpokládat, že $\alpha = \frac{a+b\sqrt{D}}{c}$, kde $a, b, c \in \mathbb{Z}$, $c > 0$ a $\mathrm{NSD}(a, b, c) = 1$. Protože je prvek α konjugovaný s prvkem $\overline{\alpha} = \frac{a-b\sqrt{D}}{c}$, dostáváme, že

$$\Delta_F = \mathrm{disc}(m_\alpha) = (\alpha - \overline{\alpha})^2 = (2b\sqrt{D}/c)^2 = 4b^2D/c^2 \in \mathbb{Z}.$$

Navíc

$$T_F(\alpha) = \alpha + \overline{\alpha} = \frac{2a}{c} \in \mathbb{Z}, \quad N_F(\alpha) = \alpha\overline{\alpha} = \frac{a^2 - b^2D}{c^2} \in \mathbb{Z}.$$

Zřejmě $c|2$, protože jinak kvůli stopě $\mathrm{NSD}(a, c) > 1$ a kvůli normě a bezčtvercovosti D by byl $\mathrm{NSD}(a, b, c)$ větší než jedna.

Rozeberme nyní případ $D \not\equiv 1 \pmod 4$. Pokud $c = 2$, tak $\Delta_F = b^2D$. Kvůli vyjádření normy pak ale musí být a i b liché (jinak by byl $\mathrm{NSD}(a, b, c) > 1$), tedy platí

$$1 \equiv a^2 \equiv b^2D \equiv D \pmod 4,$$

což je spor. Proto je $c = 1$. Dále platí, že

$$4b^2D = \Delta_F = \operatorname{disc}(m_{\sqrt{D}}) = (\sqrt{D} - (-\sqrt{D}))^2 = 4D,$$

takže i $b = 1$ a tvrzení platí.

Nyní tedy $D \equiv 1 \pmod 4$. Označíme-li si $\beta = \frac{1+\sqrt{D}}{2}$, pak jistě

$$m_\beta(x) = x^2 - x + (1 - D)/4.$$

Z toho plyne, že $\beta \in \mathbb{Z}_F = \mathbb{Z}[\alpha]$, což není možné pro $c = 1$, protože pak by $\frac{1+\sqrt{D}}{2} \in \mathbb{Z}[a + b\sqrt{D}]$. Proto je $c = 2$ a kvůli nesoudělnosti a, b, c a vyjádření normy jsou a a b lichá. Přepišme $\frac{a+b\sqrt{D}}{2}$ jako

$$\frac{a-b}{2} + b\frac{1+\sqrt{D}}{2},$$

z čehož dostaneme, že $\mathbb{Z}_F = \mathbb{Z}[\alpha] = \mathbb{Z}[\frac{1+\sqrt{D}}{2}] = \mathbb{Z}[\beta]$. Z toho plyne, že $\{1, \beta\}$ je celistvá báze, a tedy

$$\Delta_F = \operatorname{disc}(m_\beta) = \left(\frac{1+\sqrt{D}}{2} - \left(-\frac{1+\sqrt{D}}{2}\right)\right)^2 = D.$$

$\boxtimes$

Abychom sloučili oba případy předchozího tvrzení, můžeme si pomoci následující definicí:

Definice 2.24: Celé nenulové číslo Δ nazveme *fundamentálním diskriminantem*, pokud $\Delta \equiv 1 \pmod 4$ a Δ je beze čtverců, nebo je Δ dělitelné čtyřmi, $\Delta/4 \equiv 2, 3 \pmod 4$ a $\Delta/4$ je beze čtverců. Pak pro $F = \mathbb{Q}[\sqrt{\Delta}]$ je

$$\Delta_F = \Delta,$$

$$\mathbb{Z}_F = \mathbb{Z}\left[\frac{\Delta+\sqrt{\Delta}}{2}\right].$$

Použité zdroje

Všechna tvrzení a důkazy této kapitoly pochází z učebnic [30], [42] a z přednášky Komutativní okruhy, NALG100.

3. Třídová grupa imaginárních kvadratických těles

V této kapitole nás budou zajímat ideály okruhu celistvých čísel $\mathbb{Z}_F$ pro F imaginární kvadratické těleso.

Tvrzení 3.1: Nenulový ideál I okruhu celistvých čísel $\mathbb{Z}_F$ číselného tělesa F stupně d je volná abelovská podgrupa $\mathbb{Z}_F$ ranku d.

Důkaz: Ideál I je zjevně podgrupa volné grupy ranku d, tedy je to také volná podgrupa ranku $r < d$. Ukažme, že její rank je d. Zvolme $\{\beta_1, \ldots, \beta_d\}$ celistvou bázi a dále $0 \neq \alpha \in I$. Prvky $\{\alpha\beta_1, \ldots, \alpha\beta_d\}$ jsou určitě $\mathbb{Q}$-lineárně nezávislé, protože samotné β_i nezávislé byly.

Zajisté lze vyjádřit

$$\alpha\beta_j = \sum_{i=1}^{r} q_{i,j}\beta_i$$

pro každé j; koeficienty $q_{i,j}$ jsou celá čísla. Předpokládejme, že je možné, aby $r < d$. Pak by existovala pro každé i netriviální kombinace

$$\sum_{j=1}^{d} q_{i,j}c_j = 0$$

s koeficienty c_j v $\mathbb{Z}_F$. Protože ale

$$\sum_{i=1}^{r} q_{i,j}c_j\beta_i = c_j\alpha\beta_j,$$

sečtením všech rovností máme

$$0 = \sum_{i=1}^{r}\sum_{j=1}^{d} q_{i,j}c_j\beta_i = \sum_{j=1}^{d} c_j\alpha\beta_j,$$

což je ve sporu s nezávislostí $a\beta_i$. Tedy rank I je roven d.

$\boxtimes$

Předchozí tvrzení nám umožňuje zavést následující definici:

Definice 3.2: Buď F číselné těleso a I ideál v $\mathbb{Z}_K$. Potom *normou* ideálu I myslíme

$$N(I) = |\mathbb{Z}_K/I|.$$

Protože obě $\mathbb{Z}_K$ i I jsou volné grupy stejného ranku, výsledný kvocient je vždy konečný a právě zavedená norma je dobře definovaná.

Definice 3.3: I je *podílový ideál* okruhu celistvých čísel $\mathbb{Z}_F$, pokud jde o nenulový pod-$\mathbb{Z}_F$-modul F takový, že pro něj existuje nenulové $\alpha \in \mathbb{Z}_F$, aby αI byl ideál $\mathbb{Z}_F$. V případě kvadratických těles stačí uvažovat $\alpha \in \mathbb{Z}$, protože je-li αI ideál, také $\overline{\alpha}\alpha I$ je ideál pro $\overline{\alpha}$ konjugovaný prvek k α, a $\overline{\alpha}\alpha \in \mathbb{Z}$ pro $\alpha \in \mathbb{Z}_F$.

Pokud je $I \subseteq \mathbb{Z}_K$, pak je zřejmě I podílový ideál právě tehdy, když je to ideál. Takovým podílovým ideálům budeme říkat *celočíselné ideály*.

Celočíselný ideál I nazveme *primitivní*, pokud neexistuje žádné $n \in \mathbb{Z}$, aby I/n byl celočíselný ideál.

Je-li F číselné těleso, je okruh $\mathbb{Z}_F$ Dedekindův obor. Důkaz tohoto vcelku známého faktu můžete nalézt například v kapitole III českých skript [12] nebo například v kapitole 3.1 učebnice [30]. Zde pouze shrneme důsledky v následujícím tvrzení:

Tvrzení 3.4: Je-li F číselné těleso, tvoří podílové ideály abelovskou multiplikativní grupu s neutrálním prvkem $\mathbb{Z}_K$. Pro I, J dva podílové ideály je

$$IJ = \{ij \mid i \in I, j \in J\},$$

$$I^{-1} = \{x \mid x \in F, xI \subseteq \mathbb{Z}_K\}.$$

Navíc se každý podílový ideál I jednoznačně (až na pořadí) rozkládá na součin prvoideálů a pomocí tohoto faktu je jednoduché dokázat, že pro dva podílové ideály I a J platí

$$N(IJ) = N(I)N(J).$$

⊠

Reprezentace ideálů imaginárních těles a operace nad nimi

Protože budeme chtít pracovat s ideály, musíme nalézt nějakou vhodnou reprezentaci, která bude dostatečně kompaktní a zároveň v ní budeme umět rozumně počítat. V celé sekci budeme předpokládat, že F je číselné těleso s fundamentálním diskriminantem D.

Tvrzení 3.5: Každý celočíselný ideál I okruhu celistvých čísel $\mathbb{Z}_K = \mathbb{Z}[\omega]$ můžeme zapsat jako

$$a\mathbb{Z} + (b + c\omega)\mathbb{Z},$$

kde $a, b, c \in \mathbb{Z}$, $0 \leq b < a$ a c dělí a i b.

Důkaz: Obecně je

$$I = (a_1 + d_1\omega)\mathbb{Z} + (b_1 + c_1\omega)\mathbb{Z}.$$

Pokud jsou obě d_1 i c_1 nenulové, opakovaným odečítáním generátorů dostaneme

$$I = a\mathbb{Z} + (b_2 + c_2\omega)\mathbb{Z}.$$

Zřejmě je pak $a \neq 0$. Nyní k druhému generátoru přičteme vhodný násobek prvního, čímž získáme

$$I = a\mathbb{Z} + (b + c\omega)\mathbb{Z},$$

kde $0 \leq b < a$.

Uvažujme $a\omega$. Protože $a \in I$, také $a\omega \in I$, takže ho můžeme zapsat pomocí generátorů I jako

$$a\omega = x \cdot a + y \cdot (b + c\omega).$$

Porovnáním koeficientů ω dostaneme $a = y \cdot c$, tedy $c|a$. Pokud uvažujeme předchozí rovnost modulo a, dostaneme

$$0 \equiv \frac{a}{c}(b + c\omega) \equiv \frac{a}{c}b.$$

To je ale možné jenom tehdy, když $c|b$.

⊠

Předchozí tvrzení nám umožní reprezentovat celočíselné ideály jako

$$I = c(a\mathbb{Z} + (b + \omega)\mathbb{Z}).$$

Zřejmě I je primitivní právě když $c = 1$.

Pozorování 3.6: Je-li $I = a\mathbb{Z} + (b + c\omega)\mathbb{Z}$ ideál $\mathbb{Z}_F$, tak jeho norma je $N(I) = ac$. To plyne ihned z toho, že $\mathbb{Z}_F = \mathbb{Z} + \omega\mathbb{Z}$.

Tvrzení 3.7: $\mathbb{Z}$-modul

$$I = a\mathbb{Z} + \frac{-b + \sqrt{D}}{2}\mathbb{Z}$$

s $b \equiv D \pmod 2$ je primitivní celočíselný ideál $\mathbb{Z}_F$ právě tehdy, když $4a|D - b^2$.

Důkaz: Nejprve si označme $\omega = \frac{-b+\sqrt{D}}{2}$ a uvědomme si, že $\mathbb{Z}_K = \mathbb{Z}[\omega]$. I je ideál právě tehdy, když je uzavřený na násobení. Tedy I je ideál právě tehdy, když pro každé $e + f\omega$ je $(e + f\omega)I \subseteq I$. Tedy

$$(e + f\omega)I = ea\mathbb{Z} + e\omega\mathbb{Z} + fa\mathbb{Z} + f\omega^2\mathbb{Z}.$$

Pokud rozepíšeme

$$\omega^2 = \frac{b^2 - 2b\sqrt{D} + D}{4} = \frac{b^2 - 2b(-b + \sqrt{D}) - 2b^2 + D}{4} = \frac{D - b^2}{4} - b\omega,$$

dostaneme, že

$$(e + f\omega)I = ea\mathbb{Z} + fa\mathbb{Z} + \frac{D - b^2}{4}\mathbb{Z} + e\omega\mathbb{Z} - b\omega\mathbb{Z}.$$

Tedy $(e + f\omega)I \subseteq I$ právě tehdy, když $a|\frac{D-b^2}{4}$, tj. právě když $4a|D - b^2$.

$\boxtimes$

Tvrzení 3.8: Mějme ideál okruhu celistvých čísel

$$I = a\mathbb{Z} + \frac{-b + \sqrt{D}}{2}\mathbb{Z}.$$

Pak inverze I je

$$I^{-1} = \mathbb{Z} + \frac{b + \sqrt{D}}{2a}\mathbb{Z}.$$

Důkaz: Víme, že $I^{-1} = \{z \in F | zI \subseteq \mathbb{Z}_F\}$. Protože $a \in I$, vidíme, že takové z musí být tvaru $\frac{x+y\sqrt{D}}{2a}$ pro $x, y \in \mathbb{Z}$ takové, že $x \equiv yD \pmod 2$. Z toho, že $\frac{-b+\sqrt{D}}{2} \in I$ dále plyne, že

$$bx \equiv yD \pmod{2a}, \quad x \equiv by \pmod{2a} \text{ a } \quad \frac{Dy - bx}{2a} \equiv \frac{D(x - by)}{2a} \pmod 2.$$

Nyní lze již přímočaře ověřit, že $I^{-1} = \mathbb{Z} + \frac{b+\sqrt{D}}{2a}\mathbb{Z}$.

$\boxtimes$

Tvrzení 3.9: Mějme dva ideály okruhu celistvých čísel

$$I_k = a_k\mathbb{Z} + \frac{-b_k + \sqrt{D}}{2}\mathbb{Z}.$$

Označíme-li si $s = (b_1 + b_2)/2$ a $d = \mathrm{NSD}(a_1, a_2, d)$ s $ua_1 + va_2 + ws = d$, je součin I_1I_2 roven

$$I = d\left(A\mathbb{Z} + \frac{-B + \sqrt{D}}{2}\mathbb{Z}\right),$$

kde

$$A = \frac{a_1a_2}{d^2} \quad \text{a} \quad B = b_2 + \frac{a_2v(b_1-b_2)+w(D-b_2^2)/2}{d}.$$

Důkaz: Ideál I je jako $\mathbb{Z}$-modul generován generátory

$$a_1a_2, \quad \frac{-a_1b_2+a_1\sqrt{D}}{2}, \quad \frac{-a_2b_1+a_2\sqrt{D}}{2} \quad \text{a} \quad \frac{(b_1b_2+D)/2-s\sqrt{D}}{2}.$$

Přitom ale víme, že I jde zapsat jako

$$I = C\left(A\mathbb{Z} + \frac{-B+\sqrt{D}}{2}\mathbb{Z}\right).$$

Konstanta C je evidentně nejmenší koeficient u $\sqrt{D}/2$, a tedy $C = \mathrm{NSD}(a_1,a_2,s) = d$. Dále víme, že $N(I) = AC^2$. Přitom ale $N(I) = N(I_1)N(I_2) = a_1a_2$, takže opravdu je $A = a_1a_2/d^2$. Konečně, je-li $ua_1 + va_2 + ws = d$, z konstanty u $\sqrt{D}/2$ máme, že

$$B = \frac{ua_1b_2 + va_2b_1 + w\frac{b_1b_2+D}{2}}{d},$$

což můžeme upravit na

$$B = \frac{db_2 + va_2b_1 - va_2b_2 + w\frac{b_1b_2+D}{2} - wsb_2}{d} = b_2 + \frac{a_2v(b_1-b_2)+w(D-b_2^2)/2}{d}.$$

⊠

Ideály imaginárních těles jako pozitivní kvadratické formy

Směřujeme k zavedení důležitého isomorfismu mezi podílovými ideály a kvadratickými formami. Tento isomorfismus bude hrát podstatnou roli jak při reprezentaci třídové grupy, tak v algoritmech operací v třídové grupě.

Definice 3.10: Funkci $f : \mathbb{R}^2 \to \mathbb{R}$ nazveme *kvadratickou formou,* je-li tvaru

$$f(x,y) = ax^2 + bxy + cy^2,$$

kde a, b, c jsou celá čísla. Značit ji budeme jako (a,b,c). *Diskriminant* kvadratické formy je

$$\mathrm{disc}((a,b,c)) = b^2 - 4ac.$$

Kvadratická forma je *primitivní*, pokud $\mathrm{NSD}(a,b,c) = 1$. Dvě formy f a g jsou *ekvivalentní*, pokud existuje matice $\left(\begin{smallmatrix}\alpha & \beta \\ \gamma & \delta\end{smallmatrix}\right) \in \mathrm{SL}_2(\mathbb{Z})$, tj. celočíselná matice s jednotkovým determinantem, že $g(x,y) = f(\alpha x + \beta y, \gamma x + \delta y)$. Přímočarým dosazením jde ověřit, že tato ekvivalence zachovává diskriminant.

Pozorování 3.11: Je-li D fundamentální diskriminant, je každá kvadratická forma diskriminantu D primitivní. Kdyby totiž nebyla, tak $a = da_0, b = db_0, c = dc_0$ a diskriminant této formy by byl $d^2(b_0^2 - 4a_0c_0) = D$. Fundamentální diskriminant je ale buď bezčtvercový tvaru $4k+1$, což zjevně není možné, nebo je dělitelný čtyřmi, tento podíl je bezčtvercový a dává po dělení čtyřmi zbytek 2 nebo 3. V tomto případě musí být $d = 2$. Pak ale $b_0^2 - 4a_0c_0$ má dávat po dělení čtyřmi zbytek 2 nebo 3, což zjevně neplatí.

Definice 3.12: Kvadratická forma f je *pozitivně definitní*, když $\forall (x,y) \in \mathbb{R}^2 \setminus (0,0)$ je $f(x,y) > 0$, a *negativně definitní*, pokud $\forall (x,y) \in \mathbb{R}^2 \setminus (0,0)$ je $f(x,y) < 0$.

Množinu všech pozitivně definitních kvadratických forem fundamentálního diskriminantu D modulo akce $\mathrm{PSL}_2(\mathbb{Z})$ budeme značit $\mathcal{F}^+(D)$, kde $\mathrm{PSL}_2(\mathbb{Z})$ jsou matice $\mathrm{SL}_2(\mathbb{Z})$, přičemž matici A a $-A$ považujeme za ekvivalentní. Tuto ekvivalenci musíme zavést proto, aby akce těchto matic zachovávaly definitnost forem.

Tvrzení 3.13: Je-li (a,b,c) kvadratická forma diskriminantu $D < 0$, tak je buď pozitivně nebo negativně definitní, a to podle toho, zda je $a > 0$ nebo $a < 0$.

Důkaz: Protože $0 > D = b^2 - 4ac$, vidíme, že $a > 0$ právě když $c > 0$. Předpokládejme tedy, že $a > 0$, důkaz pro záporné a je obdobný. Máme tedy $a > 0$ a $c > 0$. Aby forma nebyla pozitivně definitní, potřebujeme x a y, aby $ax^2 + bxy + cy^2 < 0$. Zjevně obě x a y musí být nenulové. Vydělíme-li nerovnost y^2 a označíme-li $z = \frac{x}{y}$, získáme $az^2 + bz + c < 0$. Protože ale je $b^2 = D + 4ac < 4ac$, nemá tato nerovnice žádné reálné řešení.

$\boxtimes$

Nyní můžeme uvést první ze dvou tvrzení o vztahu ideálů okruhu celistvých čísel a pozitivních kvadratických forem.

Tvrzení 3.14: Buď D fundamentální diskriminant imaginárního kvadratického tělesa F. Označme si

$$\mathcal{F}_0^+(D) = \{(a,b,c) \mid a > 0, b^2 - 4ac = D\}/\Gamma$$

množinu všech pozitivních kvadratických forem diskriminantu D modulo akce

$$\Gamma = \left\{ \begin{pmatrix} 1 & m \\ 0 & 1 \end{pmatrix} \;\middle|\; m \in \mathbb{Z} \right\},$$

tj. třídy forem, kde (a,b,c) je ekvivalentní s každou

$$\begin{pmatrix} 1 & m \\ 0 & 1 \end{pmatrix} \cdot (a,b,c) = (a, b + 2am, c + bm + am^2).$$

Dále si označme

$$\mathcal{I}_0(D) = \{\alpha \mid \alpha \text{ podílový ideál } \mathbb{Z}_{\mathbb{Q}(\sqrt{D})}\}/\mathbb{Q}$$

množinu všech podílových ideálů modulo násobení racionálním číslem, tj. třídy ideálů, kde I je ekvivalentní s každým $(q)I$ pro $q \in \mathbb{Q}$. Dále buď φ_0 zobrazení z $\mathcal{F}_0^+(D)$ do $\mathcal{I}_0(D)$ takové, že

$$\varphi_o(a,b,c) = a\mathbb{Z} + \frac{-b + \sqrt{D}}{2}\mathbb{Z},$$

a ψ_0 zobrazení z $\mathcal{I}_0(D)$ do $\mathcal{F}_0^+(D)$, pro které

$$\psi_0\left(q\left(a\mathbb{Z} + \frac{-b + \sqrt{D}}{2}\mathbb{Z}\right)\right) = \left(a, b, \frac{b^2 - D}{4a}\right) \text{ pro } a > 0.$$

Pak tvoří zobrazení φ_0 a ψ_0 inverzní bijekci množin $\mathcal{F}_0^+(D)$ a $\mathcal{I}_0(D)$.

Důkaz: Důkaz je přímočarý, protože tvrzení v současné podobě není nijak komplikované. Nejprve potřebujeme ověřit, zda jsou obě zobrazení dobře definovaná. Protože dle tvrzení (3.7) je $a\mathbb{Z} + \frac{-b+\sqrt{D}}{2}\mathbb{Z}$ ideál právě tehdy, když $4a | b^2 - D$, vidíme, že (a,b,c) je kvadratická forma diskriminantu D právě tehdy, když $a\mathbb{Z} + \frac{-b+\sqrt{D}}{2}\mathbb{Z}$

je primitivní celočíselný ideál. Tedy φ_0 zobrazuje formy na ideály a ψ_0 zobrazuje ideály na formy a obě tato zobrazení jsou na.

Dvě ekvivalentní formy (a,b,c) a $(a, b+2am, c+bm+am^2)$ zobrazí φ_0 na ideály

$$a\mathbb{Z} + \frac{-b+\sqrt{D}}{2}\mathbb{Z} \quad \text{a} \quad a\mathbb{Z} + \frac{-b+2am+\sqrt{D}}{2}\mathbb{Z},$$

které jsou zjevně totožné, takže φ_0 je dobře definované. Naopak, uvědomíme-li si, že v každé třídě $\mathcal{I}_0(D)$ je právě jeden primitivní celočíselný ideál tvaru $a\mathbb{Z} + \frac{-b+\sqrt{D}}{2}\mathbb{Z}$ s $a > 0$, je i ψ_0 dobře definované.

Nyní už stačí vědět, že složení $\varphi_0 \circ \psi_0$ a $\psi_0 \circ \varphi_0$ jsou identity, což je zjevné.

$\boxtimes$

Třídová grupa

Definice 3.15: Buď $D < 0$ fundamentální diskriminant a $F = \mathbb{Q}(\sqrt{D})$ kvadratické těleso. Všechny podílové ideály označme $\mathcal{I}_F$. Podílový ideál I je *hlavní*, pokud $I = \alpha\mathbb{Z}_K$ pro $\alpha \in F$. Všechny hlavní ideály označíme $\mathcal{P}_F$. Faktor $\mathcal{I}_F/P_F$ budeme nazývat *třídová grupa* a značit $\mathcal{C}_F$ či $\mathcal{C}(D)$. Jinak řečeno, $\mathcal{C}_F$ je faktorgrupa $\mathcal{I}_F$ určená ekvivalencí $I \sim J \Leftrightarrow \exists \alpha \in F^* : I = (\alpha)J$. Z toho ihned plyne, že v každé třídě je vždy alespoň jeden primitivní celočíselný ideál.

Definice 3.16: Buď D fundamentální diskriminant číselného tělesa F. Velikost třídové grupy $\mathcal{C}_F$ budeme značit h_F či $h(D)$.

Není bohužel znám žádný efektivní algoritmus na zjištění velikosti třídové grupy daného fundamentálního diskriminantu, přestože se třídovou grupu pokoušel charakterizovat již Carl Friedrich Gauss. Nicméně existují alespoň odhady na velikost třídové grupy. Z Brauer-Siegelova tvrzení z [25] plyne, že pro každé reálné $\varepsilon > 0$ existuje $D_\varepsilon < 0$, že pro každý fundamentální diskriminant $D < D_\varepsilon$ platí

$$\sqrt{|D|}^{1-\varepsilon} \le h(D) \le \sqrt{|D|}^{1+\varepsilon}.$$

Dále za předpokladu rozšířené Riemannovy hypotézy lze dokázat, viz [28], že limitně

$$\frac{1+o(1)}{c_2 \ln\ln|D|}\sqrt{|D|} < h(D) < (1+o(1))c_3\sqrt{|D|}\ln\ln|D|,$$

kde $c_2 = 12e^\gamma/\pi \approx 6.8$ a $c_3 = 2e^\gamma/\pi \approx 1.113$.

Po zavedení třídové grupy můžeme zesílit tvrzení o ekvivalenci ideálů a pozitivních kvadratických forem.

Tvrzení 3.17: Buď D fundamentální diskriminant. Stejně jako v předchozím tvrzení definujme zobrazení $\varphi : \mathcal{F}^+(D) \to \mathcal{I}_F$ tak, že

$$\varphi(a,b,c) = a\mathbb{Z} + \frac{-b+\sqrt{D}}{2}\mathbb{Z},$$

a ψ zobrazení z $\mathcal{I}_F$ do $\mathcal{F}^+(D)$ tak, že

$$\psi_0\left(q\left(a\mathbb{Z} + \frac{-b+\sqrt{D}}{2}\mathbb{Z}\right)\right) = \left(a, b, \frac{b^2-D}{4a}\right) \text{ pro } a > 0.$$

Pak zobrazení φ_0 a ψ_0 tvoří inverzní bijekci množin $\mathcal{F}^+(D)$ a $\mathcal{I}_F$.

Důkaz: Díky důkazu předchozího tvrzení stačí dokázat, že zobrazení φ a ψ jsou dobře definovaná, čili že obrazy ekvivalentních prvků jsou ekvivalentní. To lze provést například přímo (a velmi pracně), což zde provádět nebudeme. Jednodušší důkaz, na který nemáme dost teorie, je možné najít v kapitole 5.2 knihy [9].

$\boxtimes$

Díky tomuto tvrzení můžeme odteď zaměňovat ideály a pozitivně definitní kvadratické formy. Pomocí této bijekce můžeme zavést následující definici, která nám umožní jednoznačně reprezentovat prvky třídové grupy:

Definice 3.18: Pozitivní kvadratickou formu (a, b, c) fundamentálního diskriminantu $D < 0$ nazveme *redukovanou*, jestliže $|b| \leq a \leq c$, a je-li navíc $|b| = a$ nebo $a = c$, pak ještě musí platit $b \geq 0$.

Tvrzení 3.19: Buď $D < 0$ fundamentální diskriminant. Pak v každé třídě třídové grupy existuje právě jedna redukovaná pozitivní kvadratická forma diskriminantu D.

Důkaz: Nejprve dokážeme existenci. Ze všech forem (a, b, c) ve zvolené třídě vyberme takovou, jejíž a je minimální. Pak je jistě $c \geq a$, protože (a, b, c) je ekvivalentní $(c, -b, a)$ (změna (x, y) za $(-y, x)$). Dále změna (x, y) za $(x + ky, y)$ pro

$$k = \left\lfloor \frac{a-b}{2a} \right\rfloor$$

nezmění hodnotu a a upraví hodnotu b do rozsahu $(-a, a]$. Taková forma už je redukovaná kromě případu, kdy $c = a$ a $b < 0$. Pak ale opět můžeme substitucí (x, y) za $(-y, x)$ změnit formu z (a, b, a) na $(a, -b, a)$.

Ještě potřebujeme ukázat jednoznačnost. Předpokládejme, že (a, b, c) je redukovaná a dokažme, že její hodnota a je minimální mezi všemi ekvivalentními formami. Každou ekvivalentní formu dostaneme akcí matice $\left(\begin{smallmatrix} m & * \\ n & * \end{smallmatrix}\right)$ na formu (a, b, c). Výsledek takové akce má koeficient $a' = am^2 + bmn + cn^2$ pro m a n celá čísla. Rozepišme si tento koeficient jako

$$am^2 + bmn + cn^2 = am^2 \left(1 + \frac{b}{a}\frac{n}{m}\right) + cn^2 = am^2 + cn^2 \left(1 + \frac{b}{c}\frac{m}{n}\right).$$

Pokud je $|n| < |m|$, použijeme druhý výraz, jinak třetí, a vidíme, že $a' \geq a$, protože $|b| \leq a \leq c$. Dále pokud má být $a' = a$, musí být $m = 1$ a $n = 0$ (nebo $m = 0$ a $n = 1$ pro případ $a = c$, ale v tom případě to ošetří podmínka $b \geq 0$), takže v dané třídě existuje jediná redukovaná pozitivní kvadratická forma.

$\boxtimes$

Na ideálech máme zadefinovanou grupovou operaci, proto si zavedeme její ekvivalent na kvadratických formách:

Definice 3.20: Buď $f_1 = (a_1, b_1, c_1)$ a $f_2 = (a_2, b_2, c_2)$ dvě kvadratické formy stejného fundamentálního diskriminantu a označme $s = (b_1 + b_2)/2$ a $d = \mathrm{NSD}(a_1, a_2, s)$ pro $ua_1 + va_2 + ws = d$. *Složením* kvadratických forem $f_1 f_2$ označujeme formu

$$f_1 f_2 = \left(\frac{a_1 a_2}{d^2}, b_2 + a_2 \frac{v(b_1 - b_2) - 2wc_2}{d}, \frac{b_3^2 - D}{4a_3}\right).$$

Použité zdroje

Tvrzení z této kapitoly jsou inspirována knihou [9], i když některá (jednodušší) tvrzení z této knihy nepochází; odhady na řád třídové grupy pochází z [25] a [28].

Důkazy tvrzení (3.1) až (3.7), (3.11) a (3.13) pochází přímo od autora, ostatní jsou (někdy ve zjednodušené formě) převzaté z [9].

4. Algoritmy pro práci s třídovou grupou

Víme, že pro fundamentální diskriminant $D < 0$ má třídová grupa řádově $\sqrt{|D|}$ prvků. Tyto prvky budeme reprezentovat jako jednoznačné redukované pozitivně definitní kvadratické formy $(a, b, (b^2 - D)/4a)$, které budeme značit jako (a, b). V celé kapitole bude $D < 0$ fundamentální diskriminant a $F = \mathbb{Q}(\sqrt{D})$ imaginární kvadratické číselné těleso.

Nejprve by se nám hodilo znát nějaké omezení na velikost koeficientů prvků třídové grupy.

Tvrzení 4.1: Buď (a, b, c) pozitivně definitní kvadratická forma diskriminantu D. Pak

(1) pokud je forma redukovaná, tak platí

$$a \leq \sqrt{|D|/3}.$$

(2) forma je redukovaná, pokud platí, že

$$a < \sqrt{|D|/4} \quad \text{a současně} \quad -a < b \leq a.$$

Důkaz: Pokud je forma redukovaná, tak z nerovností $|b| \leq a \leq c$, plyne

$$|D| = 4ac - b^2 \geq 4a^2 - a^2 = 3a^2,$$

což dokazuje bod (1). Naopak, pokud platí bod (2), tak

$$c = \frac{b^2 + |D|}{4a} \geq \frac{|D|}{4a} > \frac{a^2}{a} = a,$$

čili forma (a, b, c) je z definice redukovaná.

$\boxtimes$

Prvky třídové grupy můžeme tedy reprezentovat jako pozitivně definitní kvadratické formy (a, b), kde $0 < a \leq \sqrt{|D|/3}$ a $|b| < a$ nebo $b = a$.

Přímočaré algoritmy v třídové grupě

Naším cílem je algoritmicky popsat operace, které budeme v třídové grupě potřebovat. Jsou to:

- nalezení neutrálního prvku (a, b)
- nalezení inverze k prvku (a, b)
- násobení, tj. složení dvou prvků (a_1, b_1) a (a_2, b_2)
- umocnění, tj. opakované skládání prvku (a, b) se sebou samým

Nalezení neutrálního prvku je jednoduché, protože neutrální prvek odpovídá $\mathbb{Z}_F = \mathbb{Z} + \frac{D+\sqrt{D}}{2}\mathbb{Z}$, čili neutrální prvek v redukovaném tvaru je $(1, D \bmod 2)$.

K zjištění inverze prvku použijeme tvrzení (3.8), které říká, že ideál inverzní k ideálu $I = a\mathbb{Z} + \frac{1}{2}(-b + \sqrt{D})\mathbb{Z}$ je

$$I^{-1} = \mathbb{Z} + \frac{b + \sqrt{D}}{2a}\mathbb{Z}.$$

Vidíme tedy, že inverze k (a, b) je $(a, -b)$. Navíc tato inverze je již v redukovaném tvaru, kromě případu $b = a$. V tom případě si stačí uvědomit, že (a, b) je ekvivalentní $(a, b + 2a)$, z čehož dostaneme, že v případě $b = a$ je inverze k (a, b) rovna (a, b). Získáváme tedy triviální algoritmus na počítání inverze, jehož časová složitost je $\mathcal{O}(\lg b)$.

procedure Invert(a,b) : spočte inverz daného prvku třídové grupy
if $b = a$ **then return** (a, b) **else return** $(a, -b)$

Algoritmus 4.2: Inverze prvku třídové grupy

Nejkomplikovanější bude operace skládání. Ještě před ní se naučíme k zadané (ne nutně redukované) pozitivně definitní kvadratické formě (a, b, c) nalézt jí ekvivalentní redukovanou formu (α, β, γ).

V algoritmu využijeme dvě jednoduché úpravy, které se objevily v důkazu jednoznačnosti redukovaných forem:

1) (a, b, c) je ekvivalentní $(a, b - 2ak, c - bk + ak^2)$, což odpovídá akci $\left(\begin{smallmatrix}1 & -k\\ 0 & 1\end{smallmatrix}\right)$.

2) (a, b, c) je ekvivalentní $(c, -b, a)$, což odpovídá akci $\left(\begin{smallmatrix}0 & -1\\ 1 & 0\end{smallmatrix}\right)$.

Pomocí těchto úprav je jednoduché vytvořit následující algoritmus, který je velmi podobný Eukleidovu algoritmu:

procedure Reduce((a, b, c)) : spočte redukci dané formy
if $-a < b \leq a$ **and** $a < c$ **then return** (a, b, c)
do $(a, b, c) \leftarrow (c, -b, a)$
  buď $b = 2aq + r$ pro $-a < r \leq a$
  $c \leftarrow c - bq + aq^2$ (nebo ekvivalentně $c \leftarrow c - \frac{1}{2}(b + r)q$), $b \leftarrow r$
while $a > c$
if $a = c$ **and** $b < 0$ **then** $b \leftarrow -b$
return (a, b, c)

Algoritmus 4.3: Redukování kvadratické formy

Korektnost algoritmu již dokazovat nebudeme, plyne z důkazu jednoznačnosti redukované formy (3.19). Zato se budeme v následujících tvrzeních zabývat časovou složitostí tohoto algoritmu. Pojmem *redukční krok* budeme myslet jednu iteraci cyklu redukčního algoritmu. Nejprve jedno pomocné tvrzení:

Tvrzení 4.4: Buď (a, b, c) pozitivně definitní kvadratická forma diskriminantu D taková, že $-a < b \leq a$ a $a < \sqrt{|D|}$. Pak je forma (a, b, c) již redukovaná nebo je redukovaná forma (c, r, s) s $-b = 2cq + r$ pro $-c < r \leq c$, tj. forma, kterou dostaneme po jednom redukčním kroku.

Důkaz: Pokud je forma již redukovaná, není co dokazovat. Pokud není, tak $a > c$ nebo $a = c$ a $b < 0$. Pokud $a = c$, v následujícím kroku vznikne forma $(a, -b, a)$, která již redukovaná je. Tedy předpokládejme $a > c$.

Pokud je $-c < -b \leq c$, tak $q = 0$ a $(c, r, s) = (c, -b, a)$ je redukovaná. Pokud je $a > 2c$, tak $c < \sqrt{|D|/4}$, takže (c, r, s) je redukovaná podle tvrzení (4.1). Zbývá tedy případ $c < a < 2c$ a zároveň $-b \leq -c$ nebo $-b > c$.

Protože $|b| \leq a$, koeficient q v algoritmu musí být $q = \pm 1$, přičemž znamínko odpovídá znamínku $-b$. Dále tedy $s = a \mp b + c \geq c$ z $|b| \leq a$. Výsledná forma je tedy redukovaná kromě případu $s = c$. Tehdy je ale $a = \pm b$, což je možné jenom když $a = b > 0$. Pak $q = -1$ a $r = 2c - b \geq 0$, takže i v tomto případě je výsledná forma (c, r, s) redukovaná.

$\boxtimes$

Tvrzení 4.5: Počet redukčních kroků algoritmu je nejvýš

$$1 + \left\lceil \lg \left(\frac{a}{\sqrt{|D|}} \right) \right\rceil.$$

Důkaz: Nejprve si všimneme toho, že je-li $a > \sqrt{|D|}$, tak

$$c = \frac{b^2 + |D|}{4a} \leq \frac{a^2 + a^2}{4a} = \frac{a}{2}.$$

Tedy po prohození a a c se hodnota a sníží alespoň na polovinu. Proto po nejvýš $\lceil \lg(a/\sqrt{|D|}) \rceil$ krocích klesne hodnota a pod $\sqrt{|D|}$. Pak už stačí jen použít předchozí tvrzení.

$\boxtimes$

Tvrzení 4.6: Předpokládejme u následujících operací uvedené časové složitosti:

operace	časová složitost
$c = a + b$	$\mathcal{O}(\max(\lg a, \lg b))$
$c = a \cdot b$	$\mathcal{O}(\lg a \cdot \lg b)$
$c = a/b$	$\mathcal{O}(\lg c \cdot \lg b)$
$c = \mathrm{NSD}(a, b)$	$\mathcal{O}(\lg a \cdot \lg b)$

Pak pokud jsou počty bitů a, b, c omezené $\mathcal{O}(\lg |D|)$, je časová složitost redukčního algoritmu $\mathcal{O}(\lg^2 |D|)$.

Důkaz: Ve spojení s předchozím tvrzení je jasné, že složitost je omezena $\mathcal{O}(\lg^3 |D|)$. Zanalyzujeme ji přesněji. Označme (a_0, b_0, c_0) koeficenty formy na začátku algoritmu a buď (a_i, b_i, c_i) forma získaná po i redukčního krocích.

Složitost redukčního kroku je $\mathcal{O}(\lg(|b_i/a_i|) \lg(a_i))$ za jedno dělení se zbytkem plus $\mathcal{O}(\lg |D|)$ za ostatní operace. Použijme slabší odhad

$$\mathcal{O}\left(\lg \left| \frac{b_i}{a_i} \right| \lg |D| \right)$$

a uvažujme o vztahu a_i a b_i. Jistě je $|b_i| \leq a_{i-1}$. Sečtením složitosti všech kroků dostaneme

$$\mathcal{O}\left(\lg \left| \frac{b_0}{a_0} \frac{b_1}{a_1} \cdots \frac{b_z}{a_z} \right| \lg |D| \right)$$

Použitím uvedené nerovnosti dostaneme

$$\mathcal{O}\left(\lg \left| \frac{b_0}{a_0} \frac{a_0}{a_1} \cdots \frac{a_{z-1}}{a_z} \right| \lg |D| \right),$$

takže dostáváme $\mathcal{O}(\lg^2 |D|)$ použitím omezení na počáteční hodnoty b.

$\boxtimes$

Podrobnější analýzu redukčního algoritmu včetně určení multiplikativních konstant je možné najít v [6].

S pomocí redukčního algoritmu a definice složení kvadratických forem je jednoduché navrhout algoritmus pro násobení prvků třídové grupy. Připomeňme, že složení dvou forem (a_1, b_1, c_1) a (a_2, b_2, c_2) je pro $s = (b_1 + b_2)/2$, $d = \mathrm{NSD}(a_1, a_2, s)$ a $ua_1 + va_2 + ws = d$ rovno

$$(a_1, b_1, c_1) \circ (a_2, b_2, c_2) = \left(\frac{a_1 a_2}{d^2}, b_2 + \frac{a_2 v(b_1 - b_2) + w(D - b_2^2)/2}{d}, \frac{b_3^2 - D}{4a_3} \right).$$

procedure Multiply$((a_1, b_1), (a_2, b_2))$: spočte součin dvou prvků $\mathcal{C}_F$
$d_1, u, v \leftarrow \mathrm{NSD}(a_1, a_2)$, aby $ua_1 + va_2 = d_1$
$a \leftarrow a_1 a_2$, $b \leftarrow va_2(b_1 - b_2)$
if $d_1 \neq 1$ **then**
    $d_2, v, w \leftarrow \mathrm{NSD}(d_1, (b_1 + b_2)/2)$, aby $vd_1 + w(b_1 + b_2)/2 = d_2$
    $a \leftarrow a/d_2^2$, $b \leftarrow (bv + w(D - b_2^2)/2)/d_2$
$b \leftarrow b_2 + b \bmod 2a$
return Reduce$(a, b, (b^2 - D)/4a)$

Algoritmus 4.7: Násobení prvků třídové grupy

A ještě uvedeme specializovanou verzi pro mocnění na druhou:

procedure Square$((a_1, b_1))$: spočte druhou mocninu prvku grupy $\mathcal{C}_F$
$d, u, v \leftarrow \mathrm{NSD}(a_1, b_1)$, aby $ua_1 + vb_1 = d$
$a \leftarrow (a_1/d)^2$
$b \leftarrow v(D - b_1^2)/(2d)$
$b \leftarrow b_1 + b \bmod 2a$
return Reduce$(a, b, (b^2 - D)/4a)$

Algoritmus 4.8: Mocnění prvku třídové grupy na druhou

Tvrzení 4.9: Uvedené dva algoritmy fungují korektně a za předpokladu složitosti operací jako v (4.6) v čase $\mathcal{O}(\lg^2 |D|)$.

Důkaz: Korektnost algoritmů plyne ihned z definice skládání kvadratických forem. Časovou složitost stačí určit pro algoritmus na obecné skládání, protože druhý algoritmus je jeho speciální případ. Algoritmus počítá pevný počet násobení, dělení, největšího společného dělitele a jednu redukci, vše s čísly omezenými $\mathcal{O}(|D|)$, z čehož ihned plyne požadovaná časová složitost.

$\boxtimes$

Ještě zbývá vyřešit umocňování. Můžeme použít klasický algoritmus rozkladu exponentu na součet mocnin dvojky:

procedure Power$((a, b), e)$: spočte e-tou mocninu (a, b) pro $e \geq 0$
$(x, y) \leftarrow (1, D \bmod 2)$
while $e > 0$ **do**
    if $e \bmod 2 = 1$ **then** $(x, y) \leftarrow$Multiply$((x, y), (a, b))$
    $(a, b) \leftarrow$Square$((a, b))$
    $e \leftarrow e \operatorname{div} 2$
return (x, y)

Algoritmus 4.10: Umocňování prvku třídové grupy

Tento algoritmus je jistě korektní a potřebuje $\lceil \lg e \rceil$ umocňování na druhou a v nejhorším případě stejný počet násobení. Můžeme ho ale ještě vylepšit, když použijeme to, že dokážeme velmi rychle spočítat inverzi prvku.

Ve výše uvedeném algoritmu rozložíme e na $l+1$ bitů tak, že

$$e = \sum_{i=0}^{l} e_i \cdot 2^i.$$

Zkusíme nyní nalézt jinou reprezentaci e, která bude stejného tvaru, jenom koeficienty e_i budou moci nabývat hodnot 1, 0, a -1, přičemž hodnotu -1 budeme značit jako $\overline{1}$. Naším cílem bude samozřejmě to, aby výsledná reprezentace měla co nejméně nenulových koeficientů. Rozdělme si bity původního exponentu na skupiny, které neobsahují dvě a více sousedních nul. Každá taková skupina je tvaru $0(11^*0)^*1^*1$, kde hvězdička značí nula nebo více opakování. Není těžké ověřit, že každou takovou skupinu můžeme zapsat také jako $1(00^*\overline{1})^*0^*\overline{1}$, tedy následujícím způsobem:

původní reprezentace	$011\cdots1011\ldots1011\cdots11$
nová reprezentace	$100\cdots0\overline{1}00\ldots0\overline{1}00\cdots0\overline{1}$

Taková reprezentace má tu výhodu, že se nikdy nevyskytnou dva nenulové koeficienty za sebou, jinak řečeno, že nanejvýš $\lceil \lg(e)/2 \rceil$ koeficientů kódování exponentu je nenulových. Dá se dokonce dokázat, že popsané překódování obsahuje vždy nejmenší počet nenulových koeficientů, viz [29], sekce 14.7.

Nejprve nalezneme algoritmus, který pro zadané bity exponentu nalezne nové kódování pomocí hodnot 1, 0 a $\overline{1}$. Algoritmus bude procházet zadané bity od nejnižšího a bude si v proměnné c pamatovat, zda použil bit $\overline{1}$, za kterým nenásledoval bit 1. Tím dostaneme:

procedure Recode($e = (0e_{d-1}\ldots e_1e_0)_2$) : najde $f_d \ldots f_1 f_0$, $f_i \subset \{-1,0,1\}$,
aby $e = \sum_{i=0}^{d} f_i \cdot 2^i$
$c \leftarrow 0$
foreach $0 \le i \le d$ **do** $f_i \leftarrow e_i + c$, $c \leftarrow (e_i + e_{i+1} + c)$ div 2, $f_i \leftarrow f_i - 2c$
return $(f_d \ldots f_1 f_0)_2$

Algoritmus 4.11: Překódování bitů exponentu pomocí hodnot 1, 0 a -1

Pomocí toho kódování exponentu můžeme vylepšit mocnící algoritmus následujícím způsobem:

procedure SPower($(a,b), e$) : spočte e-tou mocninu (a,b) pro $e \ge 0$
$(x,y) \leftarrow (1, D \bmod 2)$, $c \leftarrow 0$
while $e > 0$ **or** $c > 0$ **do**
  $b \leftarrow c + (e \bmod 2)$, $e \leftarrow e$ div 2, $c \leftarrow (b + (e \bmod 2))$ div 2, $b \leftarrow b - 2c$
  if $b = 1$ **then** $(x,y) \leftarrow$Multiply($(x,y),(a,b)$)
  if $b = -1$ **then** $(x,y) \leftarrow$Multiply((x,y),Invert((a,b)))
  $(a,b) \leftarrow$Square((a,b))
return (x,y)

Algoritmus 4.12: Efektivnější umocňování prvku třídové grupy

Tvrzení 4.13: Algoritmus funguje korektně a použije $\lceil \lg e \rceil$ umocnění na druhou a nejvýš $\lceil \lg(e)/2 \rceil$ inverzí a násobení.

$\boxtimes$

Ve zbytku této kapitoly se zaměříme už jen na na vylepšení algoritmů pro operace ve třídové grupě.

Praktické vylepšení přímočarých algoritmů

Při násobení forem jsou velikosti koeficientů součinitelů i součinu omezeny konstantou $\sqrt{|D|}$, ale přitom se velká část pomocných výpočtů provádí s čísly o velikosti řádově $|D|$. Nicméně přeuspořádáním výpočtů a zahájením redukce už v průběhu násobení je možné dosáhnout toho, aby se většina pomocných výpočtů počítala pouze s čísly omezenými řádově $\sqrt{|D|}$. První takový algoritmus se objevil v [36]. Poté byl Atkinem v [2] vylepšen a nazván NUCOMP (a jeho speciální umocňovací varianta NUDUPL). V [34] a [23] byl algoritmus zobecněn, aby fungoval i na skládání indefinitních forem, a dokonce na skládání divizorů funkčních těles.

Oba algoritmy NUCOMP i NUDUPL nejsou asymptoticky rychlejší než popsaný algoritmus skládání. Nicméně pokud předpokládáme, že pomocné operace mají kvadratickou složitost, mohou být algoritmy NUCOMP a NUDUPL až čtyřikrát rychlejší, protože jim stačí provádět pomocné výpočty s poloviční přesností. Skutečně naměřené hodnoty se pohybují kolem dvojnásobného zrychlení (protože pomocné operace nejsou kvadratické a pomocných výpočtů je v NUCOMP a NUDUPL více), přesné hodnoty viz [23] a tabulky (7.1) až (7.3).

Ukážeme implementace obou algoritmů založené na [9]. Protože jde jenom o přeuspořádání výpočtů a prolnutí redukce se skládáním, nebudeme detailně dokazovat správnost, zájemci mohou podrobné důkazy najít v [34] a [23].

Nejprve začneme speciální verzí Eukleidova algoritmu, kterou obě implementace používají. Algoritmus bude postupovat jako při výpočtu rozšířeného Eukleidova algoritmu, který pro zadané a a b hledá d, u, v, aby $ua+vb=d$ a $d=\mathrm{NSD}(a,b)$. Náš algoritmus bude počítat pouze koeficient v a jakmile d klesne pod $L=\lfloor |D/4|^{1/4} \rfloor$, výpočet přeruší. Navíc v tu chvíli vrátí i hodnoty d a v z minulého kroku. Tedy:

procedure PartEucl(a, b, L)
$d' \leftarrow a, \quad v' \leftarrow 0, \quad d \leftarrow b, \quad v \leftarrow 1, \quad z = 0$
while $|d| > L$ **do**
  $q \leftarrow d'$ div $d, \quad r \leftarrow d'$ mod d
  $t \leftarrow v' - qv, \quad v' \leftarrow v, \quad v \leftarrow t, \quad d' \leftarrow d, \quad d \leftarrow r, \quad z \leftarrow z+1$
if z liché **then** $d \leftarrow -d, \quad v \leftarrow -v$
return (d, v, d', v')

Algoritmus 4.14: Částečný Eukleidův alg. pro NUCOMP a NUDUPL

Začneme algoritmem NUDUPL, který je speciální (ale častý a důležitý) případ NUCOMP. Tento algoritmus dostane pozitivně definitní kvadratickou formu a vrátí výsledek jejího složení se sebou samou. Tento výsledek je v naprosté většině případů téměř redukovaný, takže následná redukce skončí během jednoho nebo dvou kroků. Navíc kromě několika málo operací se všechny pomocné výpočty provádí na číslech omezených řádově $\sqrt{|D|}$.

procedure NUDUPL$((a,b,c), L = \lfloor |D/4|^{1/4} \rfloor)$: spočte $(a,b,c)^2$
$d_1, v \leftarrow GCD(a,b)$ tak, aby $vb \equiv d_1 \pmod a$
$A \leftarrow a/d_1,\;\; B \leftarrow b/d_1,\;\; C \leftarrow (-cv \bmod A)$, **if** $A < 2C$ **then** $C \leftarrow C - A$
$(d, v, d', v') \leftarrow$PartEucl$(A, C, L)$
if $v' = 0$ **then**
  $g \leftarrow (Bd + c)/d', \quad a_2 \leftarrow (d')^2, \quad c_2 \leftarrow d^2$
  $b_2 \leftarrow b + (d' + d)^2 - a_2 - c_2$ (rychlejší varianta od $b_2 \leftarrow b_2 + 2dd'$)
  $c_2 \leftarrow c_2 + gd_1$
  return (a_2, b_2, c_2)
$e \leftarrow (cv' + Bd')/A, \quad g \leftarrow (ev - B)/v', \quad b_2 \leftarrow ev + v'g$
if $d_1 > 1$ **then** $b_2 \leftarrow d_1 b_2, \quad v \leftarrow d_1 v, \quad v' \leftarrow d_1 v'$
$a_2 \leftarrow (d')^2, \quad c_2 \leftarrow d^2, \quad b_2 \leftarrow b_2 + (d' + d)^2 - a_2 - c_2$
$a_2 \leftarrow a_2 + ev', \quad c_2 \leftarrow c_2 + gv$
return (a_2, b_2, c_2)

Algoritmus 4.15: NUDUPL, skládání formy se sebou samou

Nyní ještě implementace obecného algoritmu pro skládání pozitivně definitních kvadratických forem. Stejně jako v předchozím algoritmu předpokládáme, že máme předpočítanou konstantu $L = \lfloor |D/4|^{1/4} \rfloor$.

procedure NUCOMP$((a_1, b_1, c_1), (a_2, b_2, c_2), L = \lfloor |D/4|^{1/4} \rfloor)$
: spočte složení daných forem
if $a_1 < a_2$ **then** prohoď (a_1, b_1, c_1) a (a_2, b_2, c_2)
$s \leftarrow (b_1 + b_2)/2, \quad n \leftarrow b_2 - s$
$d, v, u \leftarrow$ NSD(a_1, a_2), aby $va_1 + ua_2 = d$
if $d = 1$ **then** $A \leftarrow -un, \quad d_1 \leftarrow d$
else if $d|s$ **then** $A \leftarrow -un, \quad d_1 \leftarrow d, \quad a_1 \leftarrow a_1/d, \quad a_2 \leftarrow a_2/d, \quad s \leftarrow s/d$
else
  $d_1, u_1, v_1 \leftarrow$ NSD(s, d), aby $u_1 s + v_1 d = d_1$
  if $d_1 > 1$ **then** $a_1 \leftarrow a_1/d_1, \quad a_2 \leftarrow a_2/d_1, \quad s \leftarrow s/d_1, \quad d \leftarrow d/d_1$
  $l \leftarrow -u_1(u(c_1 \bmod d) + v(c_2 \bmod d)) \bmod d$
  $A \leftarrow -u(n/d) + l(a_1/d)$
$A \leftarrow A \bmod a_1, \quad$ **if** $a_1 < 2A$ **then** $A \leftarrow A - a_1$
$(d, v, d', v') \leftarrow$PartEucl$(a_1, A, L)$
if $v' = 0$ **then**
  $Q_1 \leftarrow a_2 d, \quad Q_2 \leftarrow Q_1 + n, \quad f \leftarrow Q_2/d', \quad g \leftarrow (ds + c_2)/d'$
  return $(a_2 d', 2Q_1 + b_2, df + gd_1)$
$b \leftarrow (a_2 d' + nv')/a_1, \quad Q_1 \leftarrow bd, \quad Q_2 \leftarrow Q_1 + n, \quad f \leftarrow Q_2/d'$
$e \leftarrow (sd' + c_2 v')/a_1, \quad Q_3 \leftarrow ev, \quad Q_4 \leftarrow Q_3 - s, \quad g \leftarrow Q_4/v'$
if $d_1 > 1$ **then** $v \leftarrow d_1 v, \quad v' \leftarrow d_1 v'$
return $(d'b + v'e, Q_1 + Q_2 + d_1(Q_3 + Q_4), df + vg)$

Algoritmus 4.16: NUCOMP, skládání dvou forem

Všechna dělení použitá v tomto algoritmu jsou celočíselná a stejně jako u NUDUPL je výsledná forma už téměř redukovaná, takže její redukce skončí ve většině případů také během jednoho nebo dvou kroků.

Asymptoticky nejrychlejší algoritmy v třídové grupě

Nyní popíšeme nejrychlejší známé algoritmy pro operace v třídové grupě. Přestože jsou tyto algoritmu asymptoticky nejrychlejší, multiplikativní konstanty způsobí, že se tyto algoritmy vyplácejí až pro diskriminanty o tisících bitů. Konkrétnější odhady je možné nalézt v [39].

Nejprve si označme $O(M(n))$ asymptotickou časovou složitost násobení dvou n-bitových čísel. Je známé, že na výpočetních modelech RAM [20] a Pointer Machine [5] je $M(n) = \mathcal{O}(n)$. V případě RAM je to jednoduchá aplikace násobení pomocí FFT, stačí si uvědomit, že stroj RAM umí pracovat s čísly o $\mathcal{O}(\lg n)$ bitech v konstantním čase. V případě Pointer Machine je důkaz komplikovanější, jeden z existujících algoritmů lze nalézt například v [24].

V případě Turingova stroje je nejlepší známý výsledek $M(n) = \mathcal{O}(n \lg n \lg\lg n)$. Popis tohoto algoritmu a odhad jeho složitosti je možné nalézt v [40].

Dále nás bude zajímat nejlepší složitost algoritmu na dělení. Algoritmus na dělení čísel musí mít složitost alespoň $\mathcal{O}(M(n))$, protože pomocí $ab = a/(1/b)$ lze násobit čísla pomocí dvou dělení. Naopak, pomocí Newtonovy iterace tvaru $x_{i+1} = x_i(2 - cx_i)$ je možné převést jedno dělení na několik násobení, jejichž celková časová složitost je omezená trojnásobkem časové složitosti násobení n-bitových čísel. Popis a analýzu tohoto algoritmu můžete nalézt například v [8].

Protože v algoritmech redukce kvadratické formy používáme Eukleidův algoritmus, musíme ještě zmínit nejlepší známou asymptotickou složitost výpočtu NSD. Nejrychlejší algoritmy fungují v čase $\mathcal{O}(M(n)\lg n)$, jeden z nich lze nalézt například v [31].

Nyní se můžeme pustit do složitosti algoritmů v třídové grupě. Začneme násobením:

Tvrzení 4.17: Násobení dvou prvků třídové grupy $\mathcal{C}_F$ lze provést pomocí jedné redukce a pomocných operací, přičemž tyto pomocné operace trvají dohromady $\mathcal{O}(M(n)\lg n)$, kde $n = \lg|D|$.

Důkaz: Pokud se podíváme na algoritmus (4.7), zjistíme, že kromě jedné redukce potřebujeme konstantně mnoho násobení, dělení a NSD čísel o nejvýše n bitech.

$\boxtimes$

Nyní zkusíme urychlit algoritmus redukce. Naším cílem by samozřejmě bylo dosáhnout složitosti $\mathcal{O}(M(n)\lg n)$, protože pak by i násobení prvků třídové grupy mělo složitost $\mathcal{O}(M(n)\lg n)$.

Kvadratické formy budeme v tomto redukčním algoritmu značit $[a, b, c]$, přičemž b bude vždy splňovat $0 \le b < 2a$. Kromě omezení na b je toto značení shodné se značením (a, b, c). Akci matice $M \in \mathrm{SL}_2(\mathbb{Z})$ na formu $[a, b, c]$ budeme značit jako $M \cdot [a, b, c]$.

Náš jednoduchý algoritmus na redukci používal jednak operaci na snížení b a jednak operaci na prohození a a c. Pokud bychom nechtěli prohazovat, můžeme používat druhou operaci na snížení b, která bude b snižovat tak, jako bychom prohodili a a c. Máme tedy dvě operace:

$$\begin{array}{rll} \text{operace L}: & [a,b,c] = L\cdot[a, b-2a, c-b+a] & L = \left(\begin{smallmatrix}1&1\\0&1\end{smallmatrix}\right) \\ \text{operace H}: & [a,b,c] = H\cdot[a-b+c, b-2c, c] & H = \left(\begin{smallmatrix}1&0\\1&1\end{smallmatrix}\right) \end{array}$$

Uvažme algoritmus, který dostane $a, b, c > 0$ a dokud je možné provést operaci L nebo H tak, aby všechny koeficienty nové formy byly nezáporné, tak je provádí.

Když navíc algoritmus chce provést operaci L nebo H, tak ji neprovede jednou, ale tolikrát, kolikrát může. Tento počet vyplyne z toho, že

$$\begin{array}{ll} [a,b,c] = L^t \cdot [a, b-2at, c-bt+at^2] & L^t = \left(\begin{smallmatrix} 1 & t \\ 0 & 1 \end{smallmatrix}\right) \\ [a,b,c] = H^t \cdot [a-bt+ct^2, b-2ct, c] & H^t = \left(\begin{smallmatrix} 1 & 0 \\ t & 1 \end{smallmatrix}\right) \end{array}$$

Stačí zvolit maximální t, aby všechny koeficienty nové formy byly nezáporné. Právě popsaný algoritmus je jistě ekvivalentní algoritmu (4.3). Nyní zavedeme obecnější verzi problému redukování kvadratické formy:

Definice 4.18: Buď $s > 0$ nějaká daná hranice a $[a,b,c]$ pozitivně definitní kvadratická forma, která splňuje $a, \frac{1}{2}b, c \geq s$. Pokud je $M \in \mathrm{SL}_2(\mathbb{Z})$, $M \geq 0$ (myšleno po složkách) a $[a,b,c] = M \cdot [\alpha, \beta, \gamma]$, tak formu $[\alpha, \beta, \gamma]$ nazveme *minimální nad* s, pokud platí:

1. $\alpha, \frac{1}{2}\beta, \gamma \geq s$
2. $\alpha - \beta + \gamma < s$ nebo ($\beta - 2\alpha < 2s$ a současně $\beta - 2\gamma < 2s$)

Je jednoduché si rozmyslet, že forma minimální nad $s = \frac{1}{2}$ je redukovaná.

Nejprve si rozmyslíme, jak velké koeficienty se v matici M mohou objevit.

Tvrzení 4.19: Pokud je $(a,b,c) = M \cdot [\alpha, \beta, \gamma]$, $M \geq 0$ a $[\alpha, \beta, \gamma]$ je minimální forma nad s, tak pro koeficienty matice

$$M = \begin{pmatrix} u & v \\ u' & v' \end{pmatrix}$$

platí

$$(u+u')^2 \leq \frac{a}{s}, \qquad (v+v')^2 \leq \frac{c}{s}, \qquad 2(u+u')(v+v') \leq \frac{b}{s}.$$

Důkaz: Požadované nerovnosti dostaneme ihned po rozepsání akce matice M

$$M \cdot [\alpha, \beta, \gamma] = [\alpha u^2 + \beta u u' + \gamma (u')^2, 2\alpha uv + \beta(uv' + u'v) + 2\gamma u'v', \alpha v^2 + \beta v v' + \gamma (v')^2]$$

a použitím $\alpha, \frac{1}{2}\beta, \gamma \geq s$.

⊠

Zkonstruujme algoritmus na hledání minimální formy nad s. Stejně jako v případě hledání redukované formy budeme na formu aplikovat operace L a H, dokud to půjde. Mějme $[a,b,c]$ splňující $a, \frac{1}{2}b, c \geq s$, $a - b + c \geq s$ a bez újmy na obecnosti $b - 2a \geq 2s$, takže je možné aplikovat operaci L. Spočtěme, kolikrát ji můžeme aplikovat. Pokud

$$[a,b,c] = L^t[\alpha = a, \beta = b - 2at, \gamma = c - bt + at^2],$$

hledáme největší možné t, aby $\beta \geq 2s$ a současně $\gamma \geq s$. Protože $\beta^2 - 4\alpha\gamma = D$ a $\alpha = a$, je druhá podmínka ekvivalentní podmínce $\beta^2 \geq D + 4as$, takže požadujeme $\beta \geq 2s$ a současně $\beta^2 \geq D + 4as$. Obdobně vyřešíme i operaci H.

Pomocí těchto nerovností můžeme sestavit algoritmus, který provede *jeden krok minimalizace nad* s.

procedure StepAboveS((a,b,c),s) : provede 1 krok minimalizace nad s
a vrátí formu $[\alpha,\beta,\gamma]$ a matici M
if $a \geq s$ **and** $b \geq 2s$ **and** $c \geq s$ **and** $a-b+c \geq s$ **and** $b-2a \geq s$ **then**
  if $D+4as < 4s^2$ **then** $r \leftarrow 2s$ **else** $r \leftarrow \lceil\sqrt{D+4as}\rceil$
  $t \leftarrow \lfloor(b-r)/2a\rfloor$
  return $((a, b-2at, c-bt+at^2), \left(\begin{smallmatrix}1 & t\\ 0 & 1\end{smallmatrix}\right))$
if $a \geq s$ **and** $b \geq 2s$ **and** $c \geq s$ **and** $a-b+c \geq s$ **and** $b-2c \geq s$ **then**
  if $D+4cs < 4s^2$ **then** $r \leftarrow 2s$ **else** $r \leftarrow \lceil\sqrt{D+4cs}\rceil$
  $t \leftarrow \lfloor(b-r)/2c\rfloor$
  return $((a-bt+ct^2, b-2ct, c), \left(\begin{smallmatrix}1 & 0\\ t & 1\end{smallmatrix}\right))$
return $((a,b,c), \left(\begin{smallmatrix}1 & 0\\ 0 & 1\end{smallmatrix}\right))$

Algoritmus 4.20: Jeden krok minimalizace nad s

Opakováním toho algoritmu bychom získali algoritmus na hledání minimální formy nad s, jeho složitost by ale byla stejná jako složitost původního algoritmu na redukci forem.

Asymptoticky rychlý algoritmus pro minimalizaci nad s

Předpokládejme, že máme formu $[a,b,c]$, $a, \frac{1}{2}b, c \geq s$. Budeme uvažovat jenom s tvaru $s = 2^m$. Dále předpokládejme, že $a,b,c < 2^{m+n}$. Pokud je n malé, stačí provést několik kroků minimalizace nad s. V opačném případě provedeme dvě rekurzivní redukce, každou přibližně $n/2$-bitovou. I pokud se v těchto rekurzivních redukcích omezíme na prvních $2n$ bitů, výsledek bude stále dost přesný, aby šel nakonec pomocí konstantního počtu kroků minimalizace nad s opravit.

procedure MinAboveS($(a,b,c),m$) : najde minimální ekv. formu nad 2^m,
vrátí tuto formu spolu s maticí M
$[\alpha,\beta,\gamma] \leftarrow [a,b,c]$, $M \leftarrow \left(\begin{smallmatrix}1 & 0\\ 0 & 1\end{smallmatrix}\right)$
if $\min(a,b,c) \geq 2^{m+2}$ **then**
  buď n minimální takové, že $a,b,c < 2^{m+n}$
  if $m \leq n$ **then** $p \leftarrow 0$, $m' \leftarrow m$
  else $p \leftarrow m+1-n$, $[a_0,b_0,c_0] \leftarrow [a \bmod 2^p, b \bmod 2^p, c \bmod 2^p]$
    $m' \leftarrow n$, $[a,b,c] \leftarrow [a \operatorname{div} 2^p, b \operatorname{div} 2^p, c \operatorname{div} 2^p]$
  $h \leftarrow m' + (n \operatorname{div} 2)$

  if $\min(a,b,c) < 2^h$ **then** $[x,y,z] \leftarrow [a,b,c]$
  else $([x,y,z],M) \leftarrow$MinAboveS($[a,b,c],h$)

  while $\max(x,y,z) \geq 2^h$ **and** $[x,y,z]$ není minimální nad $2^{m'}$ **do**
    $([x,y,z],M') \leftarrow$StepAboveS($[x,y,z],m'$), $M \leftarrow M \cdot M'$

  $([\alpha,\beta,\gamma],M') \leftarrow$MinAboveS($[x,y,z],m'$), $M \leftarrow M \cdot M'$
♡ **if** $p > 0$ **then** $[\alpha,\beta,\gamma] \leftarrow [\alpha,\beta,\gamma] \cdot 2^p + M^{-1} \cdot [a_0,b_0,c_0]$
while $[\alpha,\beta,\gamma]$ není minimální nad 2^m **do**
  $([\alpha,\beta,\gamma],M') \leftarrow$StepAboveS($[\alpha,\beta,\gamma],m$), $M \leftarrow M \cdot M'$
return $[\alpha,\beta,\gamma], M$

Algoritmus 4.21: Výpočet minimální formy nad s

Tvrzení 4.22: Uvedený algoritmus funguje korektně a pro $a, b, c < 2^{m+N}$, $m \leq 3N$ je jeho časová složitost $\mathcal{O}(M(N) \lg N)$.

Důkaz: Pro korektnost stačí dokázat, že když je $p > 0$, tak upravení hodnot na označeném řádku bude korektní, čili že nové hodnoty budou větší než 2^m a přitom v posledním cyklu bude stačit konstantní počet kroků nad s pro nalezení minimální formy. Ostatní kroky a případ $p = 0$ jsou zjevně korektní.

Předpokládejme tedy, že $p > 0$. Označme si prvky matice M a M^{-1} jako

$$M = \begin{pmatrix} u & v \\ u' & v' \end{pmatrix}, \qquad M^{-1} = \begin{pmatrix} v' & -v \\ -u' & u \end{pmatrix}$$

a dále buď $[\alpha_0, \beta_0, \gamma_0] = M^{-1} \cdot [a_0, b_0, c_0]$, takže

$$[\alpha_0, \beta_0, \gamma_0] = [(v')^2 a_0 + (u')^2 c_0 - u'v' b_0, (uv' + u'v) b_0 - 2vv' a_0 - 2uu' c_0, v^2 a_0 + u^2 c_0 - vu b_0].$$

Víme, že $[a, b, c] = M \cdot [\alpha, \beta, \gamma]$, kde $[\alpha, \beta, \gamma]$ je minimální nad $2^{m'}$, takže podle předchozího tvrzení o velikosti koeficientů v matici M dostaneme nerovnosti

$$(u+u')^2 \leq \frac{a}{s} < \frac{2^{m+n-p}}{2^{m'}} = 2^{m-p},\ (v+v')^2 \leq \frac{c}{s} < 2^{m-p},\ 2(u+u')(v+v') \leq \frac{b}{s} < 2 \cdot 2^{m+p}.$$

Kombinací těchto nerovností, faktu $a_0, b_0, c_0 < 2^p$ a vyjádření α_0, b_0, c_0 dostaneme

$$-2^m < \alpha_0, \frac{1}{2}\beta_0, \gamma_0 < 2 \cdot 2^m,$$

takže hodnoty $\alpha, \frac{1}{2}\beta, \gamma$ jsou po provedení označeného řádku alespoň $2^{m'} 2^p - 2^m = 2 \cdot 2^m - 2^m = 2^m$, což jsme chtěli. Na druhou stranu z odhadu na maximální hodnoty $\alpha_0, \beta_0, \gamma_0$ je zřejmé, že bude stačit konstantně mnoho iterací posledního cyklu algoritmu.

Ještě chceme odhadnout časovou složitost algoritmu. Analyzujme ji pro každé $m < 3N$ a každý vstup $a, b, c < 2^{m+N}$. Kromě rekurzivních volání používá algoritmus jenom konstantně mnoho aritmetických operací s čísly o délce $\mathcal{O}(N)$. Hodnota N v rekurzivních voláních klesne vždy alespoň o polovinu, takže dostáváme, že časová složitost

$$T(N) \leq c \cdot M(N) + 2 \cdot T(N/2) \leq c \sum_{i=0}^{\lceil \lg N \rceil} 2^i M(N/2^i) \leq c \cdot M(N) \lceil \lg(N) \rceil$$

je opravdu $\mathcal{O}(M(N) \lg N)$.

⊠

Rozšířený Eukleidův algoritmus jako redukce dané formy

Víme, že redukci dané formy s koeficienty o nejvýše n bitech můžeme provést v čase $\mathcal{O}(M(n) \lg n)$. Jistě by bylo zajímavé znát i nějaký dolní odhad.

Nejprve zopakujeme standardní výsledek z výpočetní složitosti. Spočítat druhou odmocninu zadaného celého čísla o n bitech lze v čase pěti násobení čísel o n bitech. Tento výsledek se opírá o to, že spočítat podíl zadaných dvou n-bitových čísel lze také provést v čase konstantně mnoha (v tomto případě tří) násobení. Pak stačí použít Newtonovu iteraci tvaru

$$x_{i+1} = \frac{x_i + c/x_i}{2}.$$

Protože chyba této aproximace kvadraticky klesá, v každém kroku bude zdvojnásobovat přesnost, s jakou počítáme, a chyba bude stále konstantní. Tedy jeden krok, ve kterém počítáme s k bity, trvá čas $\mathcal{O}(M(k))$, proto celý výpočet trvá

$$\sum_{i=0}^{\lceil \lg n \rceil} \mathcal{O}(M(2^i)) = \mathcal{O}\left(\sum_{i=0}^{\lceil \lg n \rceil} \frac{2^i}{2n} M(2n)\right) = \mathcal{O}\left(M(2n)\frac{2^{\lceil \lg n \rceil + 1} - 1}{2n}\right) = \mathcal{O}(M(n)).$$

Podrobnější analýzu včetně analýzy dělení a počítání multiplikativních konstant lze nalézt v [8].

Tvrzení 4.23: Buď x, y přirozená n-bitová čísla. Pak spočítat $\mathrm{NSD}(x, y)$ a přirozená čísla u, v, aby $ux + vy = \mathrm{NSD}(x, y)$, lze pomocí jedné redukce kvadratických forem, až na pomocný výpočet v asymptotickém čase násobení n-bitových čísel.

Důkaz: Položme $k = 2y$ a vytvořme následující kvadratickou formu:

$$(a, b, c) = (k^2 x^2 + 1, 2k^2 xy, k^2 y^2).$$

Tato forma je diskriminantu $D = -k^4$, jde tedy o pozitivně definitní formu záporného diskriminantu. Tento diskriminant sice není fundamentální, ale algoritmy pro redukci forem fundamentalitu diskriminantu nepotřebují.

Proveďme tedy redukci a získáme ekvivalentní redukovanou formu (α, β, γ), přičemž platí $\alpha \leq \sqrt{|D|/3} < k^2$. Víme, že redukovaná forma vznikne akcí nějaké matice

$$M = \begin{pmatrix} p & u \\ q & v \end{pmatrix},$$

jejíž koeficienty jsou celočíselné, její determimant je jedna, přičemž bez újmy na obecnosti je $p \geq 0$. Pro koeficient α tedy získáváme rovnost $\alpha = ap^2 + bpq + cq^2 = p^2 + k^2(px + qy)^2$, kterou můžeme upravit na

$$\frac{p^2}{k^2} + (px + qy)^2 = \frac{\alpha}{k^2} < 1.$$

Odtud z celočíselnosti dostaneme $p = \sqrt{\alpha}$ a $px + qy = 0$. Protože $pv - uq = 1$, p je nesoudělné s q a proto $p|y$. Máme tedy

$$x + q\frac{y}{p} = 0.$$

Číslo y/p jistě dělí y i x. Přitom x mohou dělit i nějaké násobky y/p, pokud to jsou faktory q. Ale protože je p a q nesoudělné, tyto násobky y/p nemohou dělit y. Tedy $d = y/p$ je $\mathrm{NSD}(x, y)$.

Použitím $p = y/d$ a $q = -x/d$ dostaneme

$$d = d \cdot \det M = dpv - dqu = d\frac{y}{d}v + d\frac{x}{d}u = ux + yv,$$

takže zbylé dva koeficienty matice M jsou hledané koeficienty lineární kombinace x a y. Použitím akce matice M získáme pro γ rovnost $u^2 + k^2(ux + vy)^2 = \gamma$, odkud máme $|u| = \sqrt{\gamma - k^2 d^2}$ a tedy $v = (1 + qu)/p$. Znaménko u u musíme bohužel odhadnout, ale to asymptoticky nevadí. Získali jsme tedy následující algoritmus:

procedure RNSD(x,y) : spočte $d = \mathrm{NSD}(x, y)$ a u, v, aby $ux + vy = d$
$(\alpha, _, \gamma) \leftarrow$Reduce$((4x^2y^2 + 1, 8xy^3, 4y^4))$
$p \leftarrow \sqrt{\alpha}, \quad d \leftarrow y/p, \quad q \leftarrow -x/d, \quad u \leftarrow \sqrt{\gamma - 4d^2y^2}$
if $p|(1+uq)$ **then return** $(d, u, (1+uq)/p)$ **else return** $(d, -u, (1-uq)/p)$

Algoritmus 4.24: Výpočet NSD **pomocí jedné redukce forem**

Tento algoritmus přesně následuje právě podaný důkaz a potřebuje jenom konstantně operací, které umíme všechny převést na násobení čísel daného počtu bitů, a jednu redukci kvadratické formy.

⊠

Použité zdroje

Implementace algoritmů a důkazy tvrzení až do (4.13) pocházejí od autora kromě tvrzení (4.1), (4.4) a (4.5), které i s důkazem pochází z [9]. Z [9] také pochází implementace algoritmů NUCOMP a NUDUPL. Asymptoticky nejrychlejší známý algoritmus na redukci dané formy a metoda počítání rozšířeného Eukleidova algoritmu pomocí redukce pochází z [39], i když implementace algoritmů a většina důkazu tvrzení (4.22) pochází od autora.

5. Kryptografická primitiva třídové grupy

Abychom mohli používat třídovou grupu imaginárních kvadratických těles v protokolech asymetrické kryptografie, musíme popsat výpočetní problémy, které dokážeme efektivně provést a nedokážeme efektivně invertovat. Tyto problémy budeme označovat jejich originálními anglickými zkratkami.

Buď $D < 0$ fundamentální diskriminant a $\mathcal{C}(D)$ třídová grupa kvadratického tělesa $\mathbb{Q}(\sqrt{D})$. Pak definujme následující výpočetní problémy:

- **problém odmocniny (IQ-RP)**: pro dané $\alpha \in \mathcal{C}(D)$ a prvočíslo $x \in \mathbb{Z}$ nedělící řád $\mathcal{C}(D)$ najít $\beta \in \mathcal{C}(D)$, aby $\beta^x = \alpha$.
- **problém řádu (IQ-OP)**: pro dané $\alpha \in \mathcal{C}(D)$ zjistit velikost podgrupy generované α, čili najít nejmenší přirozené x, že $\alpha^x = 1$.
- **problém diskrétního logaritmu (IQ-DLP)**: pro dané $\alpha, \beta \in \mathcal{C}(D)$ nalézt nejmenší přirozené x tak, aby $\beta = \alpha^x$ (případně říct, že neexistuje).
- **problém Diffie-Hellman (IQ-DHP)**: pomocí $\gamma, \alpha, \beta \in \mathcal{C}(D)$ splňujících $a = \gamma^x$, $\beta = \gamma^y$ pro nějaká přirozená čísla x a y spočítat γ^{xy}.

Tvrzení 5.1: Mezi uvedenými problémy jsou následující vztahy:

I) IQ-RP lze vyřešit pomocí IQ-OP,
II) IQ-OP lze vyřešit pomocí IQ-DLP,
III) IQ-DHP lze vyřešit pomocí IQ-DLP,
IV) faktorizaci zadaného čísla lze vyřešit pomocí IQ-OP,
V) kompletní faktorizace zadaného čísla je ekvivalentní s variantou IQ-RP, která počítá k zadaném prvku všechny druhé odmocniny.

Důkaz: Body II) a III) jsou zřejmé ihned.

Pro bod I) mějme dané $\alpha \in \mathcal{C}(D)$ a $x > 1$. Spočtěme si řád prvku α, nechť je to r. Protože x nedělí řád $\mathcal{C}(D)$, je x a r nesoudělné. Tedy pomocí Eukleidova algoritmu můžeme najít u, že $xu \equiv 1 \pmod{r}$. Pokud tedy položíme $\beta = \alpha^u$, máme

$$\beta^x = \alpha^{ux} = \alpha^{1+cr} = \alpha(\alpha^r)^c = \alpha.$$

Body IV) a V) jsou založeny na následujícím faktu: prvek $\alpha \in \mathcal{C}(D)$, jehož druhá mocnina je neutrální prvek $\mathcal{C}(D)$, nazvěme odmocninou z jedné. Pokud má diskriminant $D < 0$ N navzájem různých prvočíselných dělitelů, obsahuje třídová grupa přesně 2^{N-1} neekvivalentních odmocnin z jedné. Navíc z koeficientů reprezentantů těchto tříd (kromě triviálního řešení, tj. neutrálního prvku) lze získat netriviální faktory D. Důkaz tohoto tvrzení můžete najít například v [30], tvrzení 3.70. Pravděpodobnostní převod na IQ-OP je nyní nasnadě: mezi náhodnými prvky třídové grupy budeme hledat prvek se sudým řádem. Jakmile ho najdeme, získáme netriviální odmocninu z jedné, ze které získáme netriviální faktory D.

Na tvrzení s odmocninami z jedné je založená také faktorizační metoda od autorů Schnorr a Lenstra, takže postup získávání dělitelů diskriminantu z odmocnin z jedné a myšlenku deterministické redukce na IQ-OP můžete najít v jejich práci [38].

$\boxtimes$

Žádné další vztahy mezi uvedenými problémy nejsou v tuto dobu známé. Speciálně se neví, jestli je IQ-RP lehčí než IQ-OP, neví se, jestli je IQ-OP lehčí než IQ-DLP, a neví se, jestli je faktorizace lehčí než IQ-OP. V článku [4] uvažují autoři nad tím, proč zatím nebyl nalezen ekvivalent číselného síta pro řešení IQ-DLP, a uvádějí důvody, proč nejspíš nebude ani nalezen, z čehož by plynulo, že faktorizace je lehčí než IQ-DLP.

V následující sekci popíšeme nejrychlejší známý způsob výpočtu IQ-DLP (a tedy i ostatních uvedených problémů), variantu číselně faktorizačního algoritmu kvadratického síta. K tomu budeme potřebovat charakterizovat prvočinitele třídové grupy (prvoideály) a ukázat spojitost normy a faktorizace prvku třídové grupy. Důkaz následujícího tvrzení lze nalézt například v [30].

Tvrzení 5.2: Buď $D < 0$ fundamentální diskriminant. Pak

I) pokud pro prvočíslo p splňující $(\frac{D}{p}) \neq -1$ a jednoznačné $0 \leq b_p \leq p$, $b_p = \sqrt{D} \pmod{4p}$ zadefinujeme

$$\mathfrak{p}(p) = p\mathbb{Z} + \frac{b_p + \sqrt{D}}{2}\mathbb{Z} \text{ a } \mathfrak{p}(p)^{-1} = p\mathbb{Z} + \frac{-b_p + \sqrt{D}}{2}\mathbb{Z},$$

pak $\mathfrak{p}(p)$ a $\mathfrak{p}(p)^{-1}$ jsou prvoideály normy p a všechny prvoideály jsou právě popsaného tvaru. Navíc v případě $(\frac{D}{p}) = 1$ je $(p) = \mathfrak{p}(p)\mathfrak{p}(p)^{-1}$ a v případě $(\frac{D}{p}) = 0$ je $(p) = \mathfrak{p}(p)$.

II) je-li $\alpha = a\mathbb{Z} + \frac{b+\sqrt{D}}{2}\mathbb{Z}$ podílový ideál, jehož prvočíselný rozklad normy je

$$N(\alpha) = \prod_{i=1}^{d} p_i^{e_i},$$

pak je α ekvivalentní podílovému ideálu

$$\prod_{i=1}^{d} \mathfrak{p}(p_i)^{e_i z_i},$$

kde $z_i \in \{-1, 1\}$ splňují $b \equiv z_i b_{p_i} \pmod{2p}$.

⊠

Řešení IQ-DLP variantou kvadratického síta

Popišme nyní nejrychlejší známý algoritmus na řešení IQ-DLP, tj. na spočtení diskrétního logaritmu v třídové grupě imaginárního kvadratického tělesa. Budeme předpokládat, že čtenář zná kvadratické síto včetně variant MPQS a SIQS.

Mějme zadaný fundamentální diskriminant $D < 0$ a buďte α, γ podílové ideály $\mathbb{Z}_{\mathbb{Q}(\sqrt{D})}$. Chceme najít takové x, že α je ekvivalentní γ^x. Schválně jsme nenapsali, že $\alpha, \gamma \in \mathcal{C}(D)$, protože algoritmus bude pracovat s ideály a ne s třídami ideálů. Ekvivalenci podílových ideálů modulo hlavní ideály budeme značit jako $\sim$.

Stejně jako kvadratické síto bude mít tento algoritmus dvě fáze. V první budeme hledat relace a v druhé budeme řešit soustavu lineárních rovnic nad $\mathbb{Z}$.

Buď $p_1, \ldots, p_k$ prvních k prvočísel splňujících $(\frac{D}{p_i}) = 1$. Definujme *faktorbázi FB* jako $FB = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_k\}$. Pro $v = (v_1, \ldots, v_k) \in \mathbb{Z}^k$ definujme

$$FB^v = \prod_{i=1}^{k} \mathfrak{p}_i^{v_i}.$$

Řekneme, že v je *relace*, pokud je FB^v hlavní ideál. Relace mají následující vlastnosti:

1) je-li v relace a $a \in \mathbb{Z}$, je i av relace, protože $FB^{av} = (FB^v)^a = (\alpha)^a = (\alpha^a)$,
2) pro u, v relace je i $u + v$ relace, protože $FB^{u+v} = FB^u FB^v = (\alpha)(\beta) = (\alpha\beta)$.

Všechny relace $\Lambda = \{v \in \mathbb{Z}^k \mid FB^v \text{ je hlavní ideál}\}$ tedy tvoří podmříž $\mathbb{Z}^k$.

Následuje hlavní myšlenka algoritmu pro počítání diskrétního logaritmu:

procedure SolveDLP(α, γ) : najde (ne nutně nejmenší) x, aby $\alpha \sim \gamma^x$
Buď $FB = \{\mathfrak{p}_1, \ldots, \mathfrak{p}_k\}$ faktorbáze.
Najdeme $n > k$ relací $v_1, \ldots, v_n$.
Spočteme v_a, v_c tak, aby $\alpha \sim FB^{v_a}$ a $\gamma^{-1} \sim FB^{v_c}$.
Protože $\alpha^{-1}FB^{v_a}$ i γFB^{v_c} jsou hlavní, $(1, 0, v_g)$ i $(0, 1, v_a)$ jsou relace nad rozšířenou faktorbází $FB' = \{\gamma, \alpha^{-1}, \mathfrak{p}_1, \ldots, \mathfrak{p}_k\}$.
Buď matice $A = (\vec{v}_1 \ldots \vec{v}_k)$ a $B = \begin{pmatrix} 1 & 0 & \vec{0} \\ 0 & 1 & \vec{0} \\ \vec{v}_c & \vec{v}_a & A \end{pmatrix} = \begin{pmatrix} \vec{b} \\ B' \end{pmatrix}$.
Vyřešíme soustavu $B' \cdot \vec{y} = (1, 0, \ldots, 0)$ nad $\mathbb{Z}$.
$x \leftarrow \vec{b} \cdot \vec{y}$
return x

Algoritmus 5.3: Algoritmus pro řešení IQ-DLP

Nejprve si všimneme, že pokud $B' \cdot \vec{y} = (1, 0, \ldots, 0)$, tak pak i $B \cdot (x, \vec{y}) = (x, 1, 0, \ldots, 0)$. Nyní dokažme správnost algoritmu:

Tvrzení 5.4: Nechť sloupce B generují Λ. Pak $\alpha \sim \gamma^x$ má řešení právě tehdy, když existuje $\vec{y}$, že $B \cdot \vec{y} = (x, 1, 0, \ldots, 0)$.

Důkaz: Pokud takové $\vec{y}$ existuje, tak $z = (x, 1, 0, \ldots, 0)$ je jistě relace, protože je to lineární kombinace sloupců B, což jsou relace. Tedy $FB^z = \gamma^x \alpha^{-1}$ je hlavní ideál a tedy $\gamma^x \sim \alpha$.

Naopak má-li $\alpha \sim \gamma^x$ řešení, tak $\gamma^x \alpha^{-1}$ je hlavní ideál a $z = (x, 1, 0, \ldots, 0)$ je relace. Protože ale sloupce B generují celé Λ, z musí jít vyjádřit jako lineární kombinace sloupců B.

⊠

Algoritmus má v této podobě několik nevýhod. Jednak zatím neumíme ověřit, zda matice B již obsahuje všechny generátory Λ. Nicméně za předpokladu rozšířené Riemannovy hypotézy si můžeme pomoci tvrzením z [3]:

Tvrzení 5.5: Buď $D < 0$ fundamentální diskriminant. Pak za předpokladu rozšířené Riemannovy hypotézy tvoří všechny podílové prvoideály s normou nejvýše $6 \ln^2 |D|$ všechny generátory třídové grupy.

⊠

Takže na začátku vezmeme do faktorbáze všechny prvoideály s normou nejvýše $6 \ln^2 |D|$ a poté budeme generovat relace, dokud nezískáme všechny generátory. To, zda jsme získali již všechny generátory, lze otestovat, protože determinant Λ je $h(D)$ a za předpokladu Riemannovy hypotézy umíme najít h^* takové, že $h^*/2 < h(D) < h^*$. Detaily lze nalézt v článku [21].

Další problém algoritmu spočívá v tom, že nalezené x není nejmenší možné. Existuje i varianta tohoto algoritmu, která hledá minimální takové x. Tento algoritmus po nalezení generátorů Λ nalezne ještě generátory třídové grupy jako $\mathbb{Z}$-modulu a teprve pomocí těchto generátorů hledá diskrétní logaritmus. Tento algoritmus můžete nalézt v [22].

Aby byl námi popsaný algoritmus (5.3) funkční, musíme ještě

1) umět faktorizovat ideály do faktorbáze,
2) rychle generovat co největší množství relací,
3) nalézt řešení soustavy rovnic.

Bod 3) je totožný s případem faktorizace čísel, takže se jím zde nebudeme vůbec zabývat. Faktorizaci ideálů do faktorbáze také zvládneme lehce: stačí použít tvrzení (5.2), které říká, že stačí faktorizovat normu ideálu a z této faktorizace hned získáme faktorizaci na prvoideály.

Nyní už zbývá vyřešit pouze generování relací. Zvolme náhodný vektor $v \in \mathbb{Z}^k$ a spočtěme redukovaný ideál $\alpha \sim FB^v$. Pokud se ideál α rozkládá ve faktorbázi jako $\alpha = FB^a$, máme $FB^a \sim FB^v$, takže FB^{v-a} je hlavní ideál a $v - a$ je relace.

Takový postup ale nebude mít velkou pravděpodobnost nalezení relace. Proto do něj zapojíme prosívání. Nejprve vybereme náhodné koeficienty $v_i \in \{-1, 0, 1\}$ a spočteme redukovaný ideál

$$\alpha = FB^v = \prod_{i=1}^{k} \mathfrak{p}_i^{v_i} = a\mathbb{Z} + \frac{b + \sqrt{D}}{2}\mathbb{Z}.$$

Nyní budeme pomocí prosívání hledat nějaký ekvivalentní ideál, který se bude plně rozkládat nad faktorbází. Použijeme k tomu následující tvrzení:

Tvrzení 5.6: Je-li $\alpha \ni c = ax + \frac{b+\sqrt{D}}{2}y$ pro $x, y \in \mathbb{Z}$, tak $(c) = \alpha\beta$ a

$$N(\beta) = ax^2 + bxy + \frac{b^2 - D}{4a}y^2 = f(x, y).$$

Důkaz: Položíme $\beta = (c)\alpha^{-1}$ a vyjádřením normy ideálu (c) máme

$$N((c)) = |c\overline{c}| = \left| \left(ax + \frac{b+\sqrt{D}}{2}y\right)\left(ax + \frac{b-\sqrt{D}}{2}y\right) \right| = |a(ax^2 + bxy + cy^2)| = N(\alpha)N(\beta).$$

$\boxtimes$

Můžeme tedy prosíváním hledat taková $x, y \in \mathbb{Z}$, aby se hodnota $f(x, y)$ faktorizovala nad faktorbází. Pokud je totiž $\beta = FB^w$, máme $\beta \sim \alpha^{-1} \sim FB^{-v}$ a tedy $v + w$ je relace.

V kvadratickém sítě prosíváme většinou jenom polynomy jedné proměnné, takže můžeme zafixovat například y tak, aby byly hodnoty polynomu $F(x) = f(x, y)$ co nejlépe faktorizovatelné na nějakém intervalu $[-M, M]$.

Stejně jako v MPQS můžeme pro prosívání používat několik polynomů. Pro každý polynom je ale třeba vynásobit příslušné mocniny prvoideálů z faktorbáze a nalézt kořeny tohoto polynomu modulo každé prvočíslo ve faktorbázi. Oba tyto kroky jsou časově náročné, takže můžeme použít variantu SIQS, kterou nyní popíšeme.

Řekněme, že budeme prosívat na intervalu $[-M, M]$. Zvolíme si podmnožinu prvků z faktorbáze $Q = \{eq_1, \ldots, \mathfrak{q}_t\} \subset FB$ tak, aby

$$\prod_{i=1}^{t} N(\mathfrak{q}_i) = \prod_{i=1}^{t} q_i \approx \frac{\sqrt{|D|/2}}{M}.$$

Toto omezení normy zajistí, aby byly hodnoty prosívacího polynomu $f(x, 1)$ pro x na intervalu $[-M, M]$ co nejvhodnější. Ideály, které budeme prosívat, budou $\alpha = Q^v$, kde $v = \{-1,1\}^t$. Takových kombinací je 2^{t-1}, pokud počítáme α a α^{-1} za jednu. Všechny tyto ideály mají stejnou normu $a = N(\alpha) = \prod_{i=1}^t q_i$, ale různých hodnot b je 2^{t-1}, protože $b^2 = D \pmod{4a}$.

Abychom mohli rychle přecházet k následujícím ideálům (tj. počítat další hodnoty b a kořeny nového polynomu modulo p_i), uspořádáme si posloupnost v-ček do Grayova kódu, tj. tak, aby se dva po sobě jdoucí vektory lišily právě na jednom místě. Víme, že pro každý prvoideál $\mathfrak{q}_i = (q_i, t_{q_i})$ musí b splňovat $b \equiv t_{q_i} \pmod{2q_i}$. Pokud se tedy dva sousední vektory v liší na j-tém místě, stačí upravit tu komponentu b, která odpovídá q_j. Pokud si pro $1 \leq i \leq t$ zadefinujeme

$$B_i = (a/q_i)\Big((a/q_i)^{-1} \pmod{q_i}\Big) \pmod{a},$$

tak můžeme položit $b_1 = B_1 + \ldots + B_t$, což odpovídá vektoru $v = (1,\ldots,1)$, a pokud se v i-tém kroku změní j_i-tá souřadnice vektoru v na h_i, stačí upravit

$$b_{i+1} = b_i + 2B_{j_i}(-1)^{h_i} \pmod{a}.$$

Pokud si navíc označíme $r_{i,l}$ kořen i-tého polynomu modulo prvočíslo p_l, můžeme i kořeny počítat iterativně pomocí stejného předpisu

$$r_{i+1,l} = r_{i,l} + 2B_{j_i}(-1)^{h_i} \pmod{p_l}.$$

Volba kryptograficky vhodného diskriminantu třídové grupy

Třídová grupa je závislá na jediném parametru, na svém diskriminantu. Ten tedy musíme volit velmi opatrně, aby popsané výpočetní problémy byly opravdu těžko řešitelné v třídové grupě daného diskriminantu. V článku [19] jsou zkoumány všechny známé postupy výpočtu IQ-DLP:

- redukce na DLP v multiplikativní grupě konečných těles. Takové redukce jsou známy pouze pro nefundamentální diskriminanty. Proto budeme vždy používat jenom fundamentální diskriminanty. Vhodná volba je například $D = -p$ pro p prvočíslo, $p \equiv 3 \pmod 4$ nebo $D = -pq$ pro p, q prvočísla splňující $pq \equiv 3 \pmod 4$.
- metoda podobná $(p-1)$-faktorizačnímu algoritmu. Je možné použít ji pouze tehdy, je-li řád třídové grupy hladké číslo. Bohužel není znám žádný způsob generování diskriminantů takových, aby řád jejich třídové grupy byl/nebyl hladký. Nicméně v [19] autoři dokazují, že pravděpodobnost úspěšného útoku při náhodně zvoleném diskriminantu je zanedbatelná pro diskriminanty o řádově stovkách bitů (zmiňují konkrétně 672 bitů).
- algoritmy typu Baby-Step-Giant-Step (viz [9]). Jejich složitost je ovšem exponenciální k velikosti grupy. Protože je ale $h(D)$ řádově alespoň odmocnina z diskriminantu, předpokládáme, že tyto útoky jsou exponenciální k počtu bitů diskriminantu.
- varianta číselného síta pro řešení IQ-DLP (popsaná v minulé sekci). Jedná se o nejrychlejší známý subexponenciální útok.

Tedy při náhodné volbě diskriminantu tvaru $D = -p$ nebo $D = -pq$ pro p, q prvočísla je třeba pouze zvolit dostatečně veliký diskriminant, aby byl útok založený na kvadratickém sítu neproveditelný. Navíc volba $D = -pq$ má tu výhodu, že pokud někdo dokáže počítat diskrétní logaritmy, dokáže spočítat faktorizaci D, takže musel vyřešit faktorizační problém. Jde tedy o „přidanou“ bezpečnost oproti variantě $D = -p$.

Porovnáme nyní složitost nejlepších algoritmů na faktorizaci čísel a na řešení IQ-DLP a zkusíme najít velikosti vstupů obou problémů tak, aby časové nároky jejich řešení byly řádově stejné. Nejprve si označme

$$L_x[e, c] = \exp(c(\ln x)^e(\ln \ln x)^{1-e}).$$

Asymptoticky nejrychlejší algoritmus na faktorizaci čísel o n bitech, NFS, má složitost $L_n[\frac{1}{3}, \sqrt[3]{\frac{64}{9}} + o(1)]$. Dokázaná složitost kvadratického síta jako algoritmu na faktorizaci čísla o n bitech je $L_n[\frac{1}{2}, \sqrt{\frac{9}{8}} + o(1)]$, ale existuje hypotéza, že tato složitost je $L_n[\frac{1}{2}, 1 + o(1)]$. Algoritmus na řešení IQ-DLP nebyl zatím výslovně analyzován, ale předpokládá se, že jeho složitost je stejná jako složitost algoritmu kvadratického síta na faktorizaci čísel.

Budeme tedy předpokládat, že faktorizaci čísel umíme řešit v čase $L_n[\frac{1}{3}, \sqrt[3]{\frac{64}{9}}]$ a IQ-DLP v čase $L_n[\frac{1}{2}, 1]$, přičemž zanedbáme $o(1)$ a u IQ-DLP vezmeme pro jistotu nedokázanou nižší složitost. Z článku [19] převezmeme skutečně naměřené hodnoty – faktorizace 512 bitového čísla trvá 8000 MIPS-let, spočtení diskrétního logaritmu pro diskriminant s 220 bity trvá 0.187 MIPS-let. Časové odhady pro větší vstupy pak můžeme spočítat jako

$$t_m = t_n \frac{L_m[e, c]}{L_n[e, c]},$$

čímž vzniknou následující odhady:

n nebo $\|D\|$	očekávaný počet MIPS-let na	
	faktorizaci n	IQ-DLP v $\mathcal{C}(D)$
2^{512}	8.00×10^{3}	1.18×10^{07}
2^{1024}	5.99×10^{10}	7.82×10^{16}
2^{1536}	5.97×10^{15}	4.57×10^{26}
2^{2048}	6.98×10^{19}	2.14×10^{31}
2^{2560}	2.16×10^{23}	1.93×10^{37}
2^{3072}	2.64×10^{26}	5.32×10^{42}
2^{3584}	1.63×10^{29}	5.89×10^{47}
2^{4096}	5.87×10^{31}	3.16×10^{52}

Tabulka 5.7: Očekávaná výpočetní náročnost faktorizace a IQ-DLP

Z těchto výsledků můžeme také vydedukovat velikosti n a $|D|$, aby faktorizace a IQ-DLP byly řádově stejně výpočetně náročné. Dostaneme tak:

$\lg n$	$\lg \|D\|$
512	381
1024	687
1536	959
2048	1208
2560	1443
3072	1665
3584	1879
4096	2084

Tabulka 5.8: Stejně výpočetně náročné instance faktorizace a IQ-DLP

Generování náhodných prvků s rovnoměrným rozdělením

Většina kryptografických protokolů vyžaduje generování náhodných prvků grupy s rovnoměrným rozdělením. V případě třídové grupy to není jednoduchý problém, protože není známá ani velikost třídové grupy. Opět si ale můžeme pomoci tvrzením (5.5), že všechny generátory třídové grupy jsou mezi prvoideály s normou nejvýš $6 \ln^2 |D|$. Pomocí tohoto tvrzení můžeme vytvořit následující algoritmus:

procedure Random($\{\mathfrak{p}_1, \ldots, \mathfrak{p}_k\}$): vrací náhodný prvek $\mathcal{C}(D)$, kde $\mathfrak{p}_i$ jsou všechny prvoideály s $N(\mathfrak{p}_i) \leq 6 \ln^2 |D|$
$a \leftarrow (1, D \bmod 2)$, $(e_1, \ldots, e_k)$ buď náhodný vektor s $D < e_i < |D|$
foreach $1 \leq i \leq k$ **do** $a \leftarrow$Multiply(a,Power($\mathfrak{p}_i, e_i$))
return a

Algoritmus 5.9: Generování uniformně náhodného prvku

Lze dokázat (viz například [27]), že rozdělení tohoto generátoru náhodných prvků třídové grupy je téměř k nerozeznání od uniformního.

Popsaný algoritmus je ale velmi časově náročný, hlavně pokud je třeba v nějakém protokolu generovat mnoho náhodných prvků. V praxi se tedy používají méně výpočetně náročné varianty generování náhodných čísel. Žádné teoretické výsledky ohledně rozložení náhodných prvků nejsou známé, ale statisticky jsou rozložení prvků generovaných těmito algoritmy nerozlišitelné od uniformních.

procedure Random(n, b): vrací náhodný prvek $\mathcal{C}(D)$, musí platit $b^n > |D|$
$a \leftarrow (1, D \bmod 2)$
foreach $1 \leq k \leq n$ **do**
  do $p \leftarrow$náhodné číslo z $\{2, 3, \ldots, b\}$ **while** p není prvočíslo **or** $(\frac{D}{p}) \neq 1$
  $b_p \leftarrow \sqrt{D} \pmod{4p}$, aby $0 \leq b_p < p$
  if náhodný bit je nula **then** $\mathfrak{p} \leftarrow (p, b_p)$ **else** $\mathfrak{p} \leftarrow$Invert((p, b_p))
  $a \leftarrow$Multiply($a, \mathfrak{p}$)
return a

Algoritmus 5.10: Generování statisticky uniformního náhodného prvku

Požadavek tohoto algoritmu je, aby $b^n > |D|$. Hodnota b se většinou bere pevná, buď $2^{32} - 1$ nebo $2^{64} - 1$, aby se generovaná prvočísla vešla do registru počítače, takže hodnota n se poté stanoví jako $\lceil \lg |D| / \lg b \rceil$.

Použité zdroje

Definice výpočetních problémů a převody (5.1) byly převzaty z několika různých zdrojů. Tvrzení (5.2) pochází z [30]. Algoritmus kvadratického síta pro výpočet diskrétního logaritmu pochází z [21] a [22], byť je zde uveden jenom ve zjednodušené podobě. Diskuze volby kryptograficky vhodného diskriminantu pochází z [19], i když tabulky (5.7) a (5.8) vytvořil pomocí měření z [19] sám autor. Generování náhodných prvků je (včetně implementací) založeno na důkazech z článku [27].

6. Kryptografické protokoly v třídové grupě

Třídová grupa je komutativní grupa, ve které umíme efektivně provádět grupové operace a ve které je problém odmocňování IQ-RP, problém Diffie-Hellman IQ-DHP a problém diskrétního logaritmu IQ-DLP velmi výpočetně náročný. Tyto vlastnosti z ní činí strukturu vhodnou pro kryptografické využití.

Na druhou stranu neznáme řád této grupy. Díky tomu neumíme generovat prvky daného řádu, což bychom dokázali, kdybychom tento řád (a jeho faktorizaci) znali. Navíc některé protokoly vyžadují znalost řádu grupy, takže jejich použití v třídové grupě je komplikované až nemožné.

Protokoly zde popisované jsou všechny zavedené, takže budeme popisovat především jejich hlavní myšlenky. Ostatní detaily a různé důkazy bezpečnosti můžete nalézt vždy v uvedených odkazech.

V mnoha protokolech budeme potřebovat kryptograficky silnou hešovací funkci. Tuto funkci budeme vždy značit h a budeme předpokládat, že její výstup posloupnost 160-ti bitů, jak je to například u hešovací funkce SHA-1. Protokoly jdou samozřejmě přímočaře upravit pro případ delších hešů.

Protokoly pro výměnu klíče

I v třídové grupě je IQ-DHP výpočetně náročný, takže můžeme použít Diffie-Hellmanovu výměnu klíčů [11] přímo bez jakýchkoliv úprav:

- $\star$ Dva účastníci A a B se dohodnou na fundamentálním diskriminantu $D < 0$ a dále na náhodném prvku $\gamma \in \mathcal{C}(D)$.
- $\star$ A zvolí náhodné celé $1 < a < \sqrt{|D|}$, spočte $\alpha = \gamma^a$ a toto α pošle B.
- $\star$ B zvolí náhodné celé $1 < b < \sqrt{|D|}$, spočte $\beta = \gamma^b$ a toto β pošle A.
- $\star$ A spočte $\beta^a = \gamma^{ab}$, B spočte $\alpha^b = \gamma^{ab}$.

Protokol IQ-DH: Diffie-Hellmanova výměna klíče

Cílem tohoto protokolu je, aby obě zúčastněné strany získaly stejné tajemství, jehož konečnou hodnotu ale nemohou předem znát. Každý útočník, který by toto tajemství získal, musí dokázat vyřešit IQ-DHP.

Protokoly asymetrického šifrování

Kromě výměny klíče je možné použít IQ-DHP i k asymetrickému šifrování. Popsaný protokol je přímočaré použití DHIES z [1]:

- $\star$ A vybere fundamentální diskriminant $D < 0$, náhodný prvek $\gamma \in \mathcal{C}(D)$ a náhodné celé $1 < a < \sqrt{|D|}$. Veřejný klíč je (D, γ, γ^a), soukromý a.
- $\star$ B chce zašifrovat zprávu M, kterou bude moct rozšifrovat jenom A. B vygeneruje náhodné celé $1 < u < \sqrt{|D|}$ a pošle A jednak γ^u a jednak zprávu M zašifrovanou symetrickou šifrou s klíčem γ^{au}.
- $\star$ Pokud A obdrží γ^u, spočte si γ^{au} a použije tento prvek jako klíč k dešifrování symetricky zašifrované zprávě.

Protokol IQ-DHIES: Asymetrické šifrování pomocí problému Diffie-Hellman

Použít prvek γ^{au} jako klíč k symetrické šifře nemusí být úplně korektní, protože i když známe omezení velikostí koeficientů (a, b) prvku γ^{au}, neznáme jejich rozložení. Často se tedy jako klíč nepoužije přímo prvek γ^{au}, ale heš tohoto prvku, $h(\gamma^{au})$. Pokud navíc chceme šifrovat krátkou zprávu, můžeme použít variantu právě popsaného protokolu, která se nazývá asymetrické šifrování ElGamal [15]:

- $\star$ A vybere fundamentální diskriminant $D < 0$, náhodný prvek $\gamma \in \mathcal{C}(D)$ a náhodné celé $1 < a < \sqrt{|D|}$. Veřejný klíč je (D, γ, γ^a), soukromý a.
- $\star$ B chce zašifrovat zprávu $M \in \{0,1\}^n$, kterou může rozšifrovat jenom A. B vygeneruje náhodné celé $1 < u < \sqrt{|D|}$ a pošle A zprávu $(\gamma^u, C = M \oplus h(\gamma^{au}))$, kde $\oplus$ značí bitovou operaci XOR.
- $\star$ Když A obdrží γ^u, zašifrovanou zprávu získá jako $M = C \oplus h(\gamma^{au})$.

Protokol IQ-ElGamal: Asymetrické šifrování ElGamal

Použitá hešovací funkce může být i bezztrátové binární kódování prvku γ^{au}. Co se týče bezpečnosti tohoto protokolu, pokud bychom nepoužili hešovací funkci h, tak by prolomení tohoto protokolu znamenalo řešení problému IQ-DHP.

Také asymetrické šifrování Cramer-Shoup [10] je možno použít beze změn. Tento protokol je také založen na IQ-DHP, je o něm ale možné dokázat, že na rozdíl od variant IQ-ElGamal a IQ-DHIES bez hešování γ^{au} je tento protokol bezpečný proti útoku typu *adaptive chosen ciphertext attack*. To znamená, že i když má někdo možnost dešifrovat libovolné zprávy, nepomůže mu tato schopnost k získání soukromého klíče. Důkaz tohoto tvrzení lze nalézt v původním článku [10].

Protokoly pro digitální podpis

Protokoly pro digitální podpis jsou nejzajímavější, protože velká většina používaných protokolů digitálního podpisu vyžaduje znalost řádu grupy.

Začneme protokolem Guillou-Quisquater [18], protože ten tuto znalost nepotřebuje:

- $\star$ A vybere fundamentální diskriminant $D < 0$, náhodný prvek $\alpha \in \mathcal{C}(D)$ a náhodné celé $q < 2^{160}$. Veřejný klíč A je (D, q, α^{-q}), soukromý klíč je α.
- $\star$ Pokud chce A podepsat zprávu m, vybere náhodný prvek $\beta \in \mathcal{C}(D)$ a spočítá $s = h(m, \beta^q)$. Výsledný podpis je $(s, \omega = \beta\alpha^s)$.
- $\star$ Pro ověření podpisu je třeba, aby $\omega \in \mathcal{C}(D)$ a $h(m, w^q\alpha^{-qs}) = s$.

Protokol IQ-GQ: Podpisové schéma Guillou-Quisquater

Toto podpisové schéma je založené na problému IQ-RP, tedy na „nejslabším“ problému v třídové grupě. Nicméně i tento problém neumíme řešit lépe než pomocí převodu na diskrétní logaritmus. Dle [32] je tento protokol bezpečný proti útoku typu *existential forgery in chosen message attack*, tj. i když má útočník možnost nechat si podepsat libovolný dokument, nedokáže vytvořit nový (klidně i nesmyslný) dokument s platným podpisem. Čili protokol je možné použít i k prokazování identity podepisovatele.

Velmi zajímavé by bylo získat variantu podpisového schéma DSA [13], případně Schnorrova schématu [37]. Obě tato podobná schémata ale vyžadují znalost řádu grupy, ve které je schéma prováděno. Zrekapitulujme (pro naše účely lehce zobecněnou) podobu schématu DSA:

* $\star$ A vybere grupu G tak, aby 160-ti bitové prvočíslo q dělilo řád této grupy. Pak vybere náhodné $\gamma_0 \in G$ a položí $\gamma = \gamma_o^{|G|/q}$. Pokud je $\gamma = 1$, vybere jiné γ_0. Nakonec zvolí náhodné $a < 2^{160}$ a spočte $\alpha = \gamma^a$. Veřejný klíč je (G, q, γ, α), soukromý klíč je a.
* $\star$ Pro podepsání zprávy m vybere A náhodné $k < 2^{160}$, položí $\varrho = \gamma^k$, $r = h(\varrho)$ a
$$s = k^{-1}(h(m) + ar) \pmod{q}. \qquad (\heartsuit)$$
Podpis je (s, r).
* $\star$ Aby byl podpis platný, musí být $0 < s < q$ a $h(\gamma^{u1}\alpha^{u2}) = r$, kde
$$u_1 = s^{-1}h(m) \pmod{q} \text{ a } u_2 = s^{-1}r \pmod{q}.$$

Protokol DSA: Zobecněná podoba podpisového schématu DSA

Pokud chceme použít toto schéma v případě grupy neznámého řádu, musíme vyřešit problém generování prvku řádu q. Při generování veřejného klíče můžeme jako γ vzít náhodný prvek grupy G a doufat, že bude mít dost vysoký řád. V případě třídové grupy a heuristiky Cohen-Lenstra [9] víme, že pravděpodobnost, že prvek γ bude malého řádu, je velmi malá.

Pokud ale není γ řádu q, musíme upravit modulární redukci z řádku ($\heartsuit$). Máme v podstatě dvě možnosti: buď modulární redukci vůbec neprovedeme nebo ji provedeme modulo nějaké jiné prvočíslo, které nebude mít žádný vztah k řádu grupy G. Obě tyto možnosti byly použity v zavedených schématech.

Nejprve si popíšeme schéma, které modulární redukci vůbec neprovádí. Toto schéma pochází od Giraulta [16] a bylo dále vylepšeno autory Poupard a Stern [35], takže je nazýváno GPS:

* $\star$ A vybere fundamentální diskriminant $D < 0$, dále náhodný $\gamma \in \mathcal{C}(D)$ a náhodné celé $a < 2^{160}$. Veřejný klíč je $(D, \gamma, \alpha = \gamma^a)$, soukromý je a.
* $\star$ Pro podepsání zprávy m vybere A náhodné $k < 2^{400}$ a položí $\varrho = \gamma^k$ a $s = -ah(m, \varrho) + k$. Podpis je (s, ϱ).
* $\star$ Při ověření podpisu je třeba zkontrolovat, že $-2^{320} < s < 2^{400} + 2^{320}$, $\varrho \in \mathcal{C}(D)$ a $\gamma^s \alpha^{h(m,\varrho)} = \varrho$.

Protokol IQ-GPS: Podpisové schéma Girault-Poupard-Stern

Tento protokol je založen na problému diskrétního logaritmu IQ-DHP. Nevýhoda tohoto schématu je velký koeficient k. Tento koeficient musí být tak velký, protože dle [35] je tento protokol odolný proti napadení pouze pokud je qh/k zanedbatelné. Pokud považujeme 2^{-80} za zanedbatelné, $k = 2^{160} \cdot 2^{160} \cdot 2^{80} = 2^{400}$.

Nyní popíšeme druhé schéma podobné DSA, které provádí modulární redukci modulo prvočíslo, které nemá žádný vztah k řádu třídové grupy. Tento protokol není založený na problému diskrétního logaritmu, ale na problému odmocňování IQ-RP, takže se mu říká RDSA [7]:

- $\star$ A vybere fundamentální diskriminant $D < 0$, dále náhodný $\gamma \in \mathcal{C}(D)$ a 160-ti bitové prvočíslo q. Poté zvolí A náhodné celé $a < q$ a spočte $\alpha = \gamma^a$. Veřejný klíč je (D, q, γ, α), soukromý a.
- $\star$ Pro podepsání zprávy m vybere A náhodné $k < q$ a položí $\varrho = \gamma^k$, $e = h(m, \varrho)$ a $x = k - ae$. Poté vydělí A číslo x prvočíslem q se zbytkem, takže dostane $x = ql + s$, kde $0 \leq s < q$. Podpis je $(s, \varrho, \lambda = \gamma^l)$.
- $\star$ Podpis je platný, když $0 \leq s < q$, oba $\varrho, \lambda \in \mathcal{C}(D)$ a $\gamma^s \alpha^{h(m,\varrho)} \lambda^q = \varrho$.

Protokol IQ-RDSA: Podpisové schéma RDSA

Toto schéma je efektivnější než IQ-GPS, protože používá menší exponent k. V původním článku [7] autoři dokonce dokazují, že pokud dokáže útočník uspět v útoku typu *existential forgery in chosen message attack*, tj. dokáže vytvořit nový dokument s platným podpisem pomocí podepisování libovolných dokumentů, dokáže tento útočník počítat q-té odmocniny.

Tento důkaz se bohužel ukázal být chybným, protože v článku [14] byl popsán útok, který pomocí několika málo podepsaných libovolných dokumentů dokáže získat soukromý klíč A. Tento útok si v následující sekci popíšeme.

Útok na protokol IQ-RDSA

Základní myšlenka útoku je následující: k je voleno moc malé, takže

$$l = \left\lfloor \frac{x}{q} \right\rfloor = \left\lfloor \frac{k - ae}{q} \right\rfloor = \left\lfloor \frac{-ae}{q} \right\rfloor + \varepsilon \quad \text{pro } \varepsilon \in \{0, 1\}$$

je dobrá aproximace $-ae/q$, Navíc z nerovnosti $a < q$ dostaneme, že $l \approx e$.

Hodnota l není v podpisu přímo uvedena, jenom $\lambda = \gamma^l$. Pokud bychom ale dokázali zvolit e malé, věděli bychom, že l je omezeno $-e < l \leq 1$, takže bychom mohli vyřešit problém diskrétního logaritmu hrubou silou.

Kdybychom tedy dokázali volit malé e, můžeme spočítat hodnotu l hrubou silou a z ní získat dobrou aproximaci $-ae/q$, ze které můžeme získat několik nejvyšších bitů tajného klíče a. Pokud má totiž e řádově b_e bitů, tak z $k - ae = ql + s$ dostaneme

$$|ae + ql| = |k - s| < q,$$

takže

$$\left| a - \frac{-ql}{e} \right| < \frac{q}{e},$$

čili nejvyšších $b_e - 1$ bitů a a $-ql/e$ je shodných. Pokud už ale známe několik bitů a, můžeme v dalším kroce zvolit větší e a postupně takto získat všechny bity soukromého klíče a.

Než popíšeme podrobně celý útok, musíme vyřešit dva problémy – jednak potřebujeme umět co nejrychleji řešit problém diskrétního logaritmu, pokud víme, že je v nějakém malém rozsahu, a jednak potřebujeme přesvědčit schéma, aby použilo hodnotu e s předepsaným počtem bitů.

Počítání diskrétního logaritmu ze známého rozsahu

Chceme vyřešit následující problém: buď G abelovská multiplikativní grupa a γ a γ^l dva její prvky. Chceme najít l, přičemž víme, že $A \leq l \leq B$. Označme si $\omega = B - A$.

Triviální řešení hrubou silou potřebuje čas $\mathcal{O}(\omega)$. Dále můžeme použít metodu *baby-step-giant-step*: zapamatujeme si prvky

$$\{\gamma^A, \gamma^{A+\lfloor\sqrt{\omega}\rfloor}, \gamma^{A+2\lfloor\sqrt{\omega}\rfloor}, \ldots, \gamma^{A+(\lfloor\sqrt{\omega}\rfloor+1)\lfloor\sqrt{\omega}\rfloor}\}$$

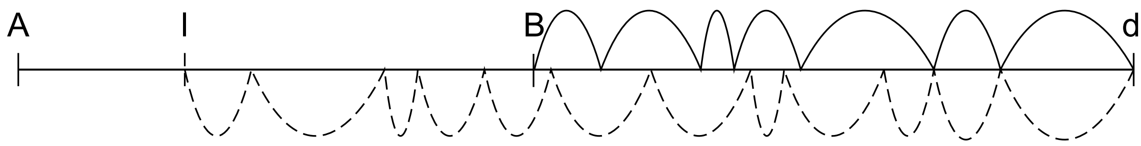
a daný prvek γ^l budeme tak dlouho násobit γ, dokud se nebude rovnat jednomu ze zapamatovaných prvků. Toto řešení má časovou i paměťovou složitost $\mathcal{O}(\sqrt{\omega})$, za předpokladu, že v zapamatovaných prvcích dokážeme vyhledávat v konstantním čase (například pomocí hešování).

Podrobněji si popíšeme ještě metodu *chytání klokanů* od Pollarda [33]. Tento algoritmus bude pravděpodobnostní (s konstantní pravděpodobností může neuspět), bude mít konstantní paměťovou složitost a časová složitost bude $\mathcal{O}(\sqrt{\omega} \lg \sqrt{\omega})$. Mějme náhodnou funkci $f : G \to S$, kde $S \subset N$ je množina několika přirozených čísel s průměrem m.

Metodu popíšeme na příkladu chytání klokana, který skáče po známé trase náhodně dlouhými skoky. Tyto skoky závisí deterministicky na místě, ze kterého se klokan odráží.

Použijeme jednoho krotkého klokana, který stáče stejně jako klokan, kterého chceme chytnout. Krotkého klokana vypustíme ze známého místa a necháme ho udělat N skoků. Tam, kde skončí, se zamaskujeme, a necháme skákat klokana, kterého chceme chytit. Pokud se dostane alespoň na jedno místo, který na své cestě navštívil krotký klokan, budou oba klokani od té doby skákat stejně a my chytíme divokého klokana na místě, kde jsme se zamaskovali.

V našem případě bude klokan skákat po různých mocninách prvku γ a délka jeho skoku bude modelovaná funkcí f. Krotkého klokana necháme skákat od prvku γ^B a necháme ho udělat N skoků. Tento klokan skončí na nějakém prvku γ^d, přičemž d bude známé. Poté začneme opět skákat od prvku γ^l, takže v každém kroku budeme znát $\gamma^{l+d'}$ a d'. Pokud po cestě narazíme na γ^d, stačí položit $l = d - d'$. Pokud navíc $A + d' > d$, víme, že se divoký klokan vyhnul nastražené pasti a pokus neuspěl.



Shrňme popsaný postup do algoritmu:

procedure SolveDLP$(\gamma, \gamma^l, N, A, B)$: najde $A \leq l \leq B$ nebo neuspěje
$\lambda = \gamma^B, \quad d = B$
foreach $1 \leq i \leq N$ **do** $skok \leftarrow f(\lambda), \quad \lambda \leftarrow \lambda\gamma^{skok}, \quad d \leftarrow d + skok$
$\lambda' = \gamma^l, \quad d' \leftarrow 0$
while $d' \leq d - A$ **do**
    $skok \leftarrow f(\lambda'), \quad \lambda' \leftarrow \lambda'\gamma^{skok}, \quad d' \leftarrow d' + skok$
    if $\lambda = \lambda'$ **then return** $d - d'$
return failure

Algoritmus 6.1: Řešení DLP ze zadaného rozsahu

Tvrzení 6.2: Popsaný algoritmus funguje správně a s vhodnou volbou N, f, S má časovou složitost $\mathcal{O}(\sqrt{\omega} \lg \sqrt{\omega})$ operací v G, konstantní paměťovou složitost a jeho pravděpodobnost neúspěchu je asymptoticky nejvýš $e^{-\theta/2}$ pro konstantu θ.

Důkaz: To, že popsaný algoritmus vrátí korektní výsledek, pokud uspěje, je zřejmé. Nejprve spočteme, jaká je pravděpodobnost neúspěchu. Položme $N = \theta m$, jde m je průměr množiny S. Předpokládejme nyní, že všechny hodnoty γ^x pro $B \leq x \leq d$ mají stejnou pravděpodobnost, že je navštívíme při cestě λ', takže tato pravděpodobnost je $1/m$. Neuspějeme právě tehdy, když prvkem λ' ani jednou nenavštívíme ani jednu z hodnot, které λ nabývalo v průběhu. Těchto hodnot je N a pravděpodobnost, že se vyhneme jedné, je $(1 - 1/m)$. Dostaneme tedy, že pravděpodobnost neúspěchu je

$$\left(1 - \frac{1}{m}\right)^N = \left(1 - \frac{1}{m}\right)^{\theta m} \approx e^{-\theta}.$$

Tento výsledek platí tehdy, když by pravděpodobnost každého γ^x byla $1/m$. Ještě musíme vybrat vhodnou množinu S. Pokud položíme

$$S = \{1, 2, \ldots, 2m - 1\},$$

bude její průměr zajisté m. Navíc pokud začneme v γ^l a budeme skákat vždy o náhodnou hodnotu $s \in S$, je pravděpodobnost návštěvy každého γ^x pro $B \leq x \leq d$ alespoň $(2m - 1)^{-1}$. To lze dokázat indukcí podle i pro prvky γ^{l+i}:

1) pro $i = 0$ je pravděpodobnost návštěvy 1,
2) pro $1 \leq i \leq 2m - 1$ je pravděpodobnost návštěvy alespoň $(2m - 1)^{-1}$, protože s pravděpodobností $(2m - 1)^{-1}$ jsme z γ^l provedli skok o i,
3) pro $2m \leq i$ je pravděpodobnost návštěvy rovna součtu

$$\sum_{j=1}^{2m-1} (\text{pravděpodobnost návštěvy } \gamma^{l+i-j}) \cdot \frac{1}{2m - 1}.$$

   Protože je ale každá z pravděpodobností návštěvy γ^{l+i-j} alespoň $(2m-1)^{-1}$, je pravděpodobnost návštěvy γ^{l+i} alespoň $(2m-1)(2m-1)^{-1}(2m-1)^{-1} = (2m - 1)^{-1}$.

S takovou volbou S tedy získáme zaručený odhad $e^{-\theta/2}$, i když pravděpodobnost navštívení každého γ^x rychle konverguje k $1/m$ a odhad neúspěchu k $e^{-\theta}$.

Ještě musíme určit časovou složitost. Nejprve provedeme N kroků s prvkem λ a potom v průměru $\frac{1}{2}\frac{w}{m} + N$ kroků s prvkem λ'. Asymptoticky tedy $\mathcal{O}(\sqrt{\omega})$ pokud bude $m = \sqrt{\omega}$. V každém kroku musíme umocnit prvek γ na nejvýše $2m-1$, z čehož plyne odhad $\mathcal{O}(\sqrt{\omega} \lg \sqrt{\omega})$.

⊠

Při volbě m bychom ale měli brát ohled také na konstanty, které se v asymptotické složitosti neprojeví. Pokud řekneme, že $m = \alpha\sqrt{\omega}$, bude celkový počet kroků algoritmu

$$N + \left(\frac{\omega}{2m} + N\right) = 2N + \frac{\omega}{2m} = \sqrt{\omega}\left(2\alpha\theta + \frac{1}{2\alpha}\right).$$

Naším cílem je, aby obě fáze skoků byly přibližně stejně dlouhé, čili chceme $2\alpha\theta = 1/2\alpha$, z čehož dostaneme $\alpha = 1/2\sqrt{\theta}$, takže optimální volba m je $\sqrt{\omega}/2\sqrt{\theta}$ a N je pak $\sqrt{\omega}\sqrt{\theta}/2$.

V původním článku zmiňuje autor ještě variantu tohoto algoritmu, která používá množinu $S = \{2^i\}$ takovou, aby její průměr byl co nejblíž potřebnému m. Pak je možné předpočítat si hodnoty γ^{2^i}, čímž časová složitost algoritmu klesne na $\mathcal{O}(\sqrt{\omega})$ operací a paměťová stoupne na $\mathcal{O}(\lg \sqrt{\omega})$. Pravděpodobnost neúspěchu tohoto algoritmu bohužel nedokázal zatím nikdo přesně určit, i když v praxi funguje tato varianta stejně dobře jako varianta s $S = \{1, 2, \ldots, 2m-1\}$, viz původní článek [33].

Získání podpisů s předepsanou velikostí e

K úspěšnému útoku potřebujeme ještě umět získat podpis s předepsanou velikostí $e = h(m, \varrho)$. To vypadá na první pohled dost nereálně. Nicméně kombinací existujících podpisů můžeme získat „pseudopodpis," jehož součástí bude konkrétní hodnota e. K tomuto pseudopodpisu nebudeme znát původní dokument, ale ten k útoku nepotřebujeme.

Mějme n skutečných podpisů $(s_i, \varrho_i, \lambda_i)$ dokumentů m_i a označme $e_i = h(m_i, \varrho_i)$. Pro libovolné $c_1, \dots, c_n \in \mathbb{Z}$ definujme

$$\tilde{e} = \sum_{i=i}^{n} c_i e_i, \qquad \tilde{s}_0 = \sum_{i=1}^{n} c_i s_i, \qquad \tilde{s} = \tilde{s}_0 \bmod q$$

$$\tilde{\lambda} = \left(\prod_{i=1}^{n} \lambda_i^{c_i} \right) \gamma^{(\tilde{s}_0 - \tilde{s})/q}, \qquad \tilde{\varrho} = \prod_{i=1}^{n} \varrho_i^{c_i}.$$

Pak nazveme $(\tilde{e}, \tilde{s}, \tilde{\varrho}, \tilde{\lambda})$ *pseudopodpis*. Tento pseudopodpis navíc projde verifikací, pokud místo $h(\tilde{m}, \tilde{\varrho})$ použijeme $\tilde{e}$:

$$\gamma^{\tilde{s}} \alpha^{\tilde{e}} \tilde{\lambda}^q = \gamma^{\tilde{s}} \alpha^{\Sigma_{i=1}^{n} c_i e_i} \left(\prod_{i=1}^{n} \lambda_i^{c_i} \right)^q \gamma^{q(\tilde{s}_0 - \tilde{s})/q} = \gamma^{\tilde{s}_0} \prod_{i=1}^{n} (\alpha^{e_i} \lambda^q)^{c_i} =$$

$$= \prod_{i=1}^{n} (\gamma^{s_i} \alpha^{e_i} \lambda^q)^{c_i} = \prod_{i=1}^{n} \varrho_i^{c_i} = \tilde{\varrho}.$$

Nyní stačí najít vhodné koeficienty c_i, aby $\tilde{e} = \sum_{i=1}^{n} c_i e_i$ mělo požadovaný počet bitů. K tomu použijeme redukční algoritmus LLL [26]. Tento algoritmus je mimo rámec této práce, takže ho zde nebudeme popisovat, popíšeme jenom jeho použití. Mějme n čísel e_i a dané číslo E. Naším cílem je nalézt koeficienty c_i tak, aby $\tilde{e} = \sum_{i=1}^{n} c_i e_i$ bylo řádově stejné jako E. Dosáhneme toho tak, že vezmeme matici

$$M = \begin{pmatrix} e_1 & E & 0 & \cdots & 0 \\ e_2 & 0 & E & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ e_n & 0 & 0 & \cdots & E \end{pmatrix},$$

pomocí LLL najdeme její LLL-redukovanou bázi a vezmeme z ní nejkratší vektor

$$V = \left(\tilde{e} = \sum_{i=1}^{n} c_i e_i, E c_1, E c_2, \dots, E c_n \right).$$

Protože vektory v redukované bázi jsou „skoro" ortogonální, protože determinant M je $E^{n-1} e$ pro $e = \sum_{i=1}^{n} e_i$ a protože jednotlivé souřadnice vektoru V mají nejspíš stejný řád ($\tilde{e} \approx E c_i$), dostaneme

$$\sqrt{n+1} \cdot \tilde{e} \approx E^{\frac{n-1}{n}} e^{1/n} \quad \text{a} \quad \sum_{i=1}^{n} |c_i| \approx \frac{n \tilde{e}}{E},$$

z čehož za předpokladu $e_i < q$ a $n \ll q$, $n \ll E$ dostaneme odhady

$$\log \tilde{e} \approx \frac{\log q}{n} + \left(1 - \frac{1}{n} \right) \log E \quad \text{a} \quad \log \left(\sum_{i=1}^{n} |c_i| \right) \approx \frac{\log q}{n} - \frac{\log E}{n}.$$

Tyto odhady jsou založené na velmi silných předpokladech, které nemusí být splněné, ale skutečné pokusy ukazují, že následující útok založený na těchto odhadech je funkční.

Útok na RDSA

Když jsme vyřešili všechny potřebné problémy, můžeme se pustit do popisu útoku. Útok bude probíhat v cyklu, v každém kroku odhalíme několik dalších bitů soukromého klíče v pořadí od nejvyšších k nejnižším. Po kroku i označme β_i počet známých bitů a. Dále buď $a = a_i + a_i'$, kde a_i je známé a neznámé a_i' je omezeno známými konstantami $\underline{A}_i \le a_i' \le \overline{A}_i$. Na začátku je $a_0 = 0$ a $\underline{A}_0 = 2, \overline{A}_o = q - 1$.

Na začátku dalšího kroku nejprve spočteme pseudopodpis $(\tilde{e}, \tilde{s}, \tilde{\varrho}, \tilde{\lambda})$ pro dané E_{i+1}, jehož hodnotu stanovíme později. Víme, že $\tilde{k} = \sum_{j=1}^{n} c_j k_j$, kde k_j jsou neznámé. Protože platí $0 \le k_j < q$, máme

$$\underline{K}_i \le \tilde{k} \le \overline{K}_i \quad \text{pro} \quad \underline{K}_i = (q-1) \sum_{c_j<0} c_j \quad \text{a} \quad \overline{K}_i = (q-1) \sum_{c_j>0} c_j.$$

Pomocí omezení $\tilde{k}$ a a_i' dostaneme ihned odhad na $(\tilde{k} - a\tilde{e})/q$:

$$\frac{\underline{K}_i - \overline{A}_i \tilde{e}}{q} - \frac{a_i \tilde{e}}{q} - 1 < \left\lfloor \frac{\tilde{k} - (a_i + a_i')\tilde{e}}{q} \right\rfloor \le \frac{\overline{K}_i - \underline{A}_i \tilde{e}}{q} - \frac{a_i \tilde{e}}{q},$$

takže

$$\tilde{l} = \left\lfloor -\frac{a_i \tilde{e}}{q} \right\rfloor + \varepsilon, \quad \text{kde} \quad \varepsilon \in \left\{ \left\lfloor \frac{\underline{K}_i - \overline{A}_i \tilde{e}}{q} \right\rfloor, \dots, \left\lfloor \frac{\overline{K}_i - \underline{A}_i \tilde{e}}{q} \right\rfloor + 1 \right\}.$$

Pomocí popsaného algoritmu na řešení problému diskrétního logaritmu ze známého rozsahu můžeme nyní najít skutečnou hodnotu $\tilde{l}$. Protože $\tilde{k} - a\tilde{e} = q\tilde{l} + \tilde{s}$, pomocí odhadů $\tilde{k}$ získáme

$$\frac{\underline{K}_i}{\tilde{e}} - \frac{q\tilde{l} + \tilde{s}}{\tilde{e}} \le a \le \frac{\overline{K}_i}{\tilde{e}} - \frac{q\tilde{l} + \tilde{s}}{\tilde{e}},$$

takže můžeme položit

$$a_{i+1} = \left\lfloor -\frac{q\tilde{l} + \tilde{s}}{\tilde{e}} \right\rfloor, \quad \underline{A}_{i+1} = \left\lfloor \frac{\underline{K}_i}{\tilde{e}} \right\rfloor \quad \text{a} \quad \overline{A}_{i+1} = \left\lfloor \frac{\overline{K}_i}{\tilde{e}} \right\rfloor.$$

Zjistěme, kolik bitů a známe po této úpravě:

$$\beta_{i+1} = \log q - \log(\overline{A}_{i+1} - \underline{A}_{i+1}) \approx \log q - \log\left(\frac{\overline{K}_i - \underline{K}_i}{\tilde{e}}\right) \approx \log q -$$

$$- \left(\log q + \log \sum_{i=1}^{n} |c_i| - \log \tilde{e}\right) \approx \frac{\log q}{n} - \frac{\log E_{i+1}}{n} - \frac{\log q}{n} - \left(1 - \frac{1}{n}\right) \log E_{i+1} \approx \log E_{i+1}.$$

Zbývá tedy určit hodnotu E_{i+1} tak, aby $E_{i+1} > E_i$, a přitom byl problém diskrétního logaritmu z daného rozsahu řešitelný. Řekněme, že chceme použít nejvýš T operací na hledání diskrétního logaritmu. Tedy

$$T \approx \sqrt{\frac{\overline{K}_i - \underline{K}_i + \tilde{e}(\overline{A}_i - \underline{A}_i)}{q}}.$$

Protože

$$\log\left(\frac{\overline{K}_i - \underline{K}_i}{q}\right) \approx \log \sum_{i=1}^{n} |c_i| \approx \frac{\log q}{n} - \frac{\log E_{i+1}}{n},$$

$$\log\left(\frac{\tilde{e}(\overline{A}_i - \underline{A}_i)}{q}\right) \approx \log \tilde{e} + \log\left(\frac{\overline{A}_i - \underline{A}_i}{q}\right) \approx \frac{\log q}{n} + \left(1 + \frac{1}{n}\right) \log E_{i+1} - \beta_i,$$

získáváme, že

$$\log(T^2) \approx \frac{\log q}{n} + \left(1 + \frac{1}{n}\right) \log E_{i+1} - \beta_i,$$

takže

$$\beta_{i+1} \approx \log E_{i+1} \approx \frac{2 \log T + \beta_i - \frac{1}{n} \log q}{1 - \frac{1}{n}}.$$

Z této rovnosti je patrné, že čím víc známých podpisů máme, tím je útok efektivnější. V článku [14] je možné prohlédnout si hodnoty a_i při útoku na 160-bitový soukromý klíč pomocí deseti známých podpisů. Algoritmus potřeboval 16 kroků k získání celého soukromého klíče.

Existuje varianta protokolu RDSA s názvem RDSA2, která nevolí parametr $k < q$, ale volí $k < qg^2$, kde g je odhad řádu grupy G. Tato varianta tedy používá řádově podobné exponenty jako GPS, ale je založena na „slabším“ výpočetním problému a produkuje delší podpisy. Schéma GPS je tedy zřejmě výhodnější.

Použité zdroje

Popsané protokoly jsou bez větších změn převzaté z citovaných materiálů. Útok na RDSA pochází z [14]. Základní metody výpočtu diskrétních logaritmů pochází od autora, metoda chytání klokanů z [33], i když její implementace a větší část důkazu pochází také od autora.

7. Implementace knihovny primitiv a protokolů

Na základě popsaných třídových operací a kryptografických primitiv a protokolů jsem vytvořil efektivní knihovnu, která umožňuje jednoduché používání popsaných protokolů. Rozhodl jsem se, že implementuji protokoly IQ-ElGamal, IQ-GQ a IQ-GPS.

Implementaci jsem vytvořil v jazyce `C`, protože pak je možné vzniklou knihovnu využívat skoro ve všech počítačových jazycích. Pro operace s libovolně velkými čísly jsem použil knihovnu GMP [17]. Z této knihovny jsem také převzal konvenci názvů identifikátorů.

Prvky třídové grupy jsou typu `iqc_t`, přičemž tento typ obsahuje dvě celá čísla `a` a `b` typu `mpz_t`. Používaný diskriminant je globální, zjišťuje se pomocí volání `iqc_get_disc(mpz_t d)` a nastavuje voláním `iqc_set_disc(mpz_t d)`.

Operace v třídové grupě se provádí voláním funkcí:

- `iqc_identity(iqc_t x)`,
- `iqc_invert(iqc_t x, iqc_t)`,
- `iqc_mul(iqc_t x, iqc_t a, iqc_t b)`, `iqc_sqr(iqc_t x, iqc_t a)`,
- `iqc_exp(iqc_t x, iqc_t a, mpz_t e)` a
- `iqc_random(iqc_t x)`.

Násobení jsem naimplementoval jednak klasickým algoritmem `Multiply` (4.7) a jednak algoritmem `NUCOMP` (4.16). Je možné vybrat si buď konkrétní implementaci příponou `_basic` nebo `_nucomp` u relevantních funkcí nebo použít výchozí implementaci. Ta se může měnit při kompilaci knihovny. Stejným způsobem naimplementoval dva algoritmy umocňování, `Power` (4.10) s příponou `_basic` a `SPower` (4.12) s příponou `_signed`. Porovnání rychlostí těchto operací je uvedeno na konci této kapitoly.

Pro generování náhodných prvků jsem použil algoritmus (5.10). V tomto algoritmu je třeba při generování prvoideálů počítat hodnoty $D^{1/2} \pmod{4p}$, kde p je prvočíslo splňující $\left(\frac{D}{p}\right) = 1$. Ponecháme-li jednoduchý případ $p = 2$ stranou, víme, že odmocnina $b_0 \equiv \sqrt{D} \pmod{p}$ vždy existuje, dokážeme ji najít například pomocí algoritmu Shanks-Tonelli [41], a navíc víme, že $p - b_0$ je také odmocnina. Nyní potřebujeme nějakou z nich rozšířit do odmocniny mod $4p$.

Pokud je $D \equiv 0 \pmod 4$ a b_0 je sudé, pak zřejmě $b_0^2 \equiv D \pmod 4$. Stejně pokud je $D \equiv 1 \pmod 4$ a b_0 je liché, platí $b_0^2 \equiv D \pmod 4$. Protože je ovšem p liché, tak právě jedna z odmocnin $b_0, p - b_0$ je sudá a jedna lichá, takže si vždy stačí vybrat tu správnou.

Pro každý z protokolů `elgamal`, `gq` a `gps` existují typy pro jejich veřejný a soukromý klíč, které se jmenují `proto_pubk_t` a `proto_privk_t` (`proto` je zástupné slovo pro jeden z protokolů `elgamal`, `gq` a `gps`). Tyto klíče je možné vygenerovat funkcí `proto_gen(int, proto_pubk_t, proto_privk_t)`, jejíž první parametr je počet bitů diskriminantu, který se zvolí tvaru $-pq$, kde p a q jsou prvočísla splňující $pq \equiv 3 \pmod 4$. Vygenerované klíče je samozřejmě možné načítat a ukládat do souborů. Klíče ukládám z výukových důvodů všechny v čitelné podobě.

Zašifrovaná zpráva se v protokolu IQ-ElGamal předává pomocí proměnné typu `elgamal_enc_t`. Samotné šifrování protokolu IQ-ElGamal se provádí pomocí funkce `elgamal_enc(elgamal_pubk_t, data, elgamal_enc_t)`, dešifrování pomocí funkce `elgamal_dec(elgamal_pubk_t, elgamal_privk_t, elgamal_enc_t, data)`.

Podpisy ve schématech IQ-GQ a IQ-GPS jsou typu `proto_sig_t`, podepisuje se funkcí `proto_sig(proto_pubk_t, proto_privk_t, hashed_message, proto_sig_t)` a ověřuje se pomocí `proto_ver(proto_pubk_t, hashed_message, proto_sig_t)`.

Implementace všech protokolů je přímočará, přesně kopíruje popis protokolů z minulé kapitoly. Jako hešovací funkci jsem použil SHA-1.

Zdrojové kódy popsané knihovny můžete najít na přiloženém DVD nebo na internetu na adrese *http://fox.ucw.cz/papers/iqc/*. Kromě zdrojového kódu obsahuje knihovna také devět spustitelných programů,

- `elgamal_gen`, `elgamal_enc`, `elgamal_dec`,
- `gq_gen`, `gq_sig`, `gq_ver`,
- `gps_gen`, `gps_sig`, `gps_ver`,

které berou parametry z příkazové řádky, volají stejnojmenné popsané funkce a měří čas, které k provedení této funkce potřebovaly.

Pomocí těchto programů jsem změřil, jaký efekt mělo vylepšení algoritmů násobení a mocnění v třídové grupě. Pro každý protokol a pro různé velikosti diskriminantu jsem změřil potřebný počet vteřin na provedení pěti operací.

$\lg\|D\|$	Multiply+Power			Multiply+SPower			NUCOMP+Power			NUCOMP+SPower		
	gen	šif	dešif	gen	šif	dešif	gen	šif	dešif	gen	šif	dešif
381	0.09	0.13	0.06	0.08	0.12	0.06	0.08	0.11	0.06	0.07	0.10	0.05
687	0.32	0.45	0.22	0.31	0.41	0.20	0.28	0.36	0.18	0.26	0.32	0.16
959	0.75	0.93	0.46	0.69	0.84	0.42	0.65	0.70	0.36	0.60	0.64	0.32
1208	1.18	1.57	0.76	1.08	1.41	0.70	0.98	1.16	0.57	0.92	1.04	0.52
1443	1.62	2.37	1.18	1.47	2.13	1.05	1.32	1.72	0.88	1.21	1.54	0.78
1665	3.31	3.29	1.64	3.12	2.95	1.47	2.86	2.36	1.18	2.73	2.10	1.08
1879	3.83	4.47	2.21	3.52	4.03	1.97	3.15	3.10	1.57	2.95	2.76	1.39
2048	6.17	5.41	2.75	5.86	4.87	2.40	5.35	3.78	1.92	5.10	3.36	1.68

Tabulka 7.1: Doba[s] pěti generování klíčů, šifrování a dešifrování ElGamal

$\lg\|D\|$	Multiply+Power			Multiply+SPower			NUCOMP+Power			NUCOMP+SPower		
	gen	podp	ver	gen	podp	ver	gen	podp	ver	gen	podp	ver
381	0.08	0.11	0.10	0.08	0.09	0.09	0.08	0.09	0.09	0.07	0.08	0.08
687	0.22	0.20	0.20	0.20	0.18	0.18	0.21	0.17	0.16	0.19	0.15	0.14
959	0.36	0.31	0.31	0.35	0.27	0.27	0.34	0.25	0.24	0.33	0.22	0.22
1208	0.58	0.42	0.41	0.56	0.37	0.37	0.55	0.33	0.31	0.54	0.30	0.28
1443	1.00	0.51	0.51	0.97	0.45	0.46	0.96	0.40	0.38	0.93	0.36	0.34
1665	1.91	0.64	0.62	1.88	0.56	0.56	1.86	0.49	0.45	1.83	0.44	0.40
1879	1.52	0.74	0.74	1.48	0.66	0.65	1.45	0.56	0.52	1.42	0.51	0.46
2048	4.28	0.84	0.84	4.24	0.72	0.72	4.20	0.64	0.59	4.18	0.57	0.51

Tabulka 7.2: Doba[s] pěti generování klíčů, podepsání a verifikace GQ

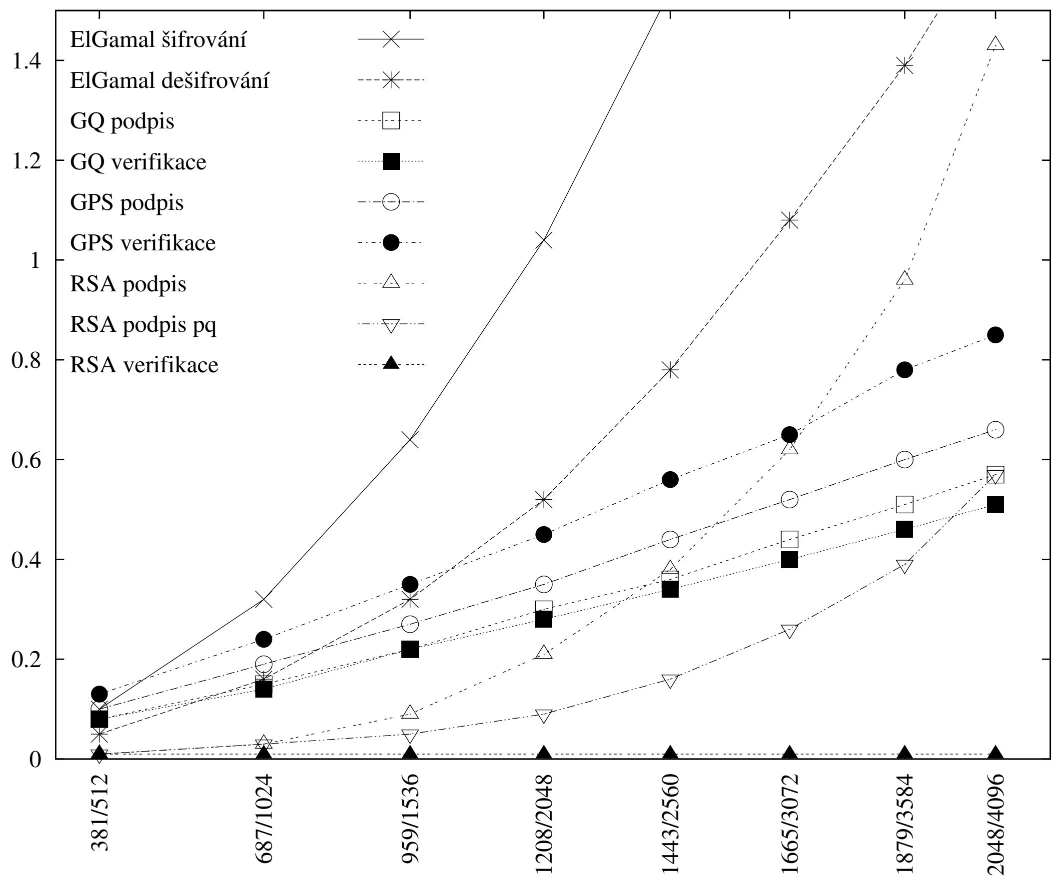
$\lg\|D\|$	Multiply+Power			Multiply+SPower			NUCOMP+Power			NUCOMP+SPower		
	gen	podp	ver	gen	podp	ver	gen	podp	ver	gen	podp	ver
381	0.08	0.13	0.18	0.08	0.12	0.16	0.08	0.12	0.15	0.07	0.10	0.13
687	0.22	0.26	0.34	0.20	0.23	0.31	0.20	0.21	0.27	0.19	0.19	0.24
959	0.36	0.38	0.50	0.34	0.35	0.46	0.34	0.30	0.38	0.32	0.27	0.35
1208	0.59	0.50	0.68	0.56	0.46	0.60	0.55	0.39	0.50	0.53	0.35	0.45
1443	0.99	0.63	0.84	0.97	0.58	0.77	0.95	0.48	0.64	0.94	0.44	0.56
1665	1.90	0.77	1.00	1.89	0.70	0.92	1.86	0.56	0.74	1.84	0.52	0.65
1879	1.52	0.91	1.22	1.48	0.82	1.10	1.44	0.66	0.86	1.42	0.60	0.78
2048	4.27	1.04	1.35	4.23	0.94	1.23	4.18	0.74	0.94	4.17	0.66	0.85

Tabulka 7.3: Doba[s] pěti generování klíčů, podepsání a verifikace GPS

Nejrychlejší implementace (používající algoritmy NUCOMP+SPower) protokolů kvadratické kryptografie jsem ještě porovnal s podpisovým schématem RSA. To jsem v rámci objektivity naimplementoval také pomocí knihovny GMP a s dvěma variantami algoritmu podpisu – bez znalosti faktorizace modulu („podp“) a se znalostí faktorizace modulu(„podp pq“). Při porovnání jsem dle (5.8) použil RSA modul s řádově stejnou složitostí útoku na privátní klíč. Výsledky uvádím v následující tabulce a v následujícím grafu:

$\lg \lvert D\rvert$	$\lg n$	ElGamal		GQ		GPS		RSA		
		šif	dešif	podp	ver	podp	ver	podp	podp pq	ver
381	512	0.10	0.05	0.08	0.08	0.10	0.13	0.01	0.01	0.01
687	1024	0.32	0.16	0.15	0.14	0.19	0.24	0.03	0.03	0.01
959	1536	0.64	0.32	0.22	0.22	0.27	0.35	0.09	0.05	0.01
1208	2048	1.04	0.52	0.30	0.28	0.35	0.45	0.21	0.09	0.01
1443	2560	1.54	0.78	0.36	0.34	0.44	0.56	0.38	0.16	0.01
1665	3072	2.10	1.08	0.44	0.40	0.52	0.65	0.62	0.26	0.01
1879	3584	2.76	1.39	0.51	0.46	0.60	0.78	0.96	0.39	0.01
2048	4096	3.36	1.68	0.57	0.51	0.66	0.85	1.43	0.57	0.01

Tabulka 7.4: Doba[s] pěti operací protokolů pomocí NUCOMP+SPower



Graf 7.5: Doba[s] pěti operací protokolů pomocí NUCOMP+SPower

Vidíme, že obě podpisová schémata IQ-GQ a IQ-GPS při velkých diskriminantech předstihnou RSA v rychlosti podepisování. Ovšem kvůli časté volbě veřejného exponentu 65537 potřebuje verifikace podpisu RSA pouhé desítky násobení, takže v rychlosti ověřování podpisu je RSA řádově nejlepší.

8. Použitá literatura

[1] Abdalla, M.,Bellare, M.,Rogaway, P.: *An encryption scheme based on the Diffie-Hellman problem,* Topics in Cryptology – CT-RSA 2001, Lecture Notes in Computer Science vol. 2020, 2001, pp. 143–158.

[2] Atkin, O.: *Letter to Dan Shanks on the programs NUDUPL and NUCOMP,* 12^{th} December 1998.

[3] Bach, E.: *Explicit bounds for primality testing and related problems,* Mathematics of Computation vol. 55, 1990, 355–380.

[4] Bauer, M. L., Hamdy, S.: *On class group computations using the number field sieve,* Lecture Notes in Computer Science vol. 2894, 2003, pp. 311–325.

[5] Ben-Amram, A.: *What is a Pointer machine,* SIGACT News (ACM Special Interest Group on Automata and Computability Theory) vol. 26, 1995.

[6] Biehl, I., Buchmann, J.: *An analysis of the reduction algorithms for binary quadratic forms,* Technical Report No. TI-26/97, Technische Universitat Darmstadt, 1997.

[7] Biehl, I., Buchmann, J., Hamdy, S., Meyer, A.: *A signature scheme based on intractability of computing roots,* Designs, Codes and Cryptography vol. 25/3, 2002, pp. 223–236.

[8] Brent, R. P.: *Multiple-precision zero-finding methods and the complexity of elementary function evaluation,* Analytic computational complexity, Proc. Sympos., Carnegie-Mellon Univ., Pittsburgh, Pa., 1975, pp. 151–176.

[9] Cohen, H.: *A course in computational algebraic number theory,* Springer, ISBN 3540556400, 1993.

[10] Cramer, R., Shoup, V.: *A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack,* Lecture Notes in Computer Science vol. 1462, 1998, pp. 13–25.

[11] Diffie, W., Hellman, M. E.: *New directions in cryptography,* IEEE Transactions and Information Theory IT-22 no. 6, 1976, pp. 644–654.

[12] Drápal, A.: *skripta k předmětu Komutativní okruhy,* http://www.karlin.mff.cuni.cz/˜drapal/komalg.ps.

[13] FIPS 186-2: *Digital Signature Standard,* NIST, Federal Information Processing Standards Publication 186-2, 2000.

[14] Fouque, P.-A., Poupard, G.: *On the security of RDSA,* Lecture Notes in Computer Science vol. 2656, 2003, pp. 643–656.

[15] Gamal, T. E.: *A public key cryptosystem and a signature scheme based on discrete logarithms,* IEEE Transactions on Information Theory 31/4, 1985, pp. 469–472.

[16] Girault, M.: *Self-certified public keys,* Lecture Notes in Computer Science vol. 547, 1992, pp. 490–497.

[17] Granlund, T. et al.: *GNU Multiple Precision Arithmetic Library,* http://www.gmplib.org/.

[18] Guillou, L. C.,Quisquater, J.-J.: *A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory,* Lecture Notes in Computer Science on Advances in Cryptology-EUROCRYPT, 1988, pp. 123–128.

[19] HAMDY, S., MÖLLER, B.: *Security of cryptosystems based on class groups of imaginary quadratic orders,* Lecture Notes in Computer Science vol. 1976, 2000, pp. 234–247.

[20] HARTMANIS, J.: *Computational complexity of random access stored program machines,* Mathematical Systems Theory 5, 1971, pp. 232–245.

[21] JACOBSON, M. J. JR: *Applying sieving to the computation of quadratic class groups,* Mathematics of Computation vol. 68, 1999, pp. 859–867.

[22] JACOBSON, M. J. JR: *Computing discrete logarithms in quadratic orders,* Journal of Cryptology vol. 13, 2000, pp. 473–492.

[23] JACOBSON, M. J. JR., VAN DER POORTEN, A. J.: *Computational aspects of NUCOMP,* Lecture Notes in Computer Science vol. 2369, 2002, pp. 185–201.

[24] KNUTH, D. E.: *The art of computer programming volume 2: Seminumerical algorithms,* Addison-Wesley, ISBN 0201896842, 1997, section 4.3.3.

[25] LANG, S.: *Algebraic number theory,* Springer, ISBN 0387942254, 1994.

[26] LENSTRA, A. K., LENSTRA, H. W. JR., LOVÁSZ, L.: *Factoring polynomials with rational coeficients,* Mathematische Annalen vol. 261/4, 1982, pp. 515–534.

[27] LENSTRA, H. W. JR. POMERANCE, C.: *A rigorous time bound for factoring integers,* Journal of American Mathematical Society vol. 5, 1992, pp. 483–516.

[28] LITTLEWOOD, J. E.: *On the class number of the corpus $P(\sqrt{-k})$,* Proceedings of the London Mathematical Society, vol. 27 of 2^{nd} series, Cambdirge University Press, 1928, pp. 358–372.

[29] MENEZES, A. J., OORSCHOT, P. C., VANSTONE, S. A.: *The Handbook of applied cryptography,* CRC Press, ISBN: 0849385237, 1997.

[30] MOLLIN, R. A.: *Algebraic number theory,* CRC Press, ISBN 0849339898, 1999.

[31] MÖLLER, N.: *On Schönhage's algorithm and subquadratic integer gcd computation,* Mathematics of Computation vol. 77, 2008, 589–607.

[32] DAVID POINTCHEVAL AND JACQUES STERN: *Security Arguments for Digital Signatures and Blind Signatures,* Journal of Cryptology: the journal of the International Association for Cryptologic Research vol. 13/3, 2000, pp. 361–396.

[33] POLLARD, J. M.: *Monte Carlo methods for index computation mod p,* Mathematics of Computation, vol. 32/143, 1978, pp. 918–924.

[34] VAN DER POORTEN, A. J.: *A note on NUCOMP,* Mathematics of Computation vol. 72, 2003, pp. 1935–1946.

[35] POUPARD, G., STERN, J.: *Security analysis of a practical "on the fly" authentication and signature generation,* Lecture Notes in Computer Science vol. 1403, 1998, pp. 422–436.

[36] SHANKS, D.: *On Gauss and composition,* Number theory and Applications, R. Mollin (ed.), Kluwer Academic Publishers, 1989, pp. 163–204.

[37] SCHNORR, C. P.: *Efficient signature generation by smart cards,* Journal of Cryptology vol. 4/3, 1991, pp. 161–174.

[38] Schnorr, C. P., Lenstra, H. W. Jr.: *A monte carlo factoring algorithm with linear storage,* Mathematics of Computation vol. 43, 1984, pp. 284–311.

[39] Schönhage, A.: *Fast reduction and composition of binary quadratic forms,* ISSAC, 1991, pp. 128–133.

[40] Schönhage, A., Strassen, V.: *Schnelle Multiplikation großer Zahlen,* Computing 7, 1971, pp. 281–292.

[41] Tornaría, G.: *Square roots modulo p,* Lecture Notes in Computer Science vol. 2286, 2002, pp. 73–86.

[42] Weiss, E.: *Algebraic number theory,* Courier Dover Publications, ISBN 0486401898, 1998.