VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
ÚSTAV MATEMATIKY

FACULTY OF MECHANICAL ENGINEERING
INSTITUTE OF MATHEMATICS

GRÖBNEROVY BÁZE, ČUANG-C'ŮV ALGORITMUS A ATAKY MULTIVARIAČNÍCH KRYPTOSYSTÉMŮ

GRÖBNER BASIS, ZHUANG-ZI ALGORITHM AND ATTACKS OF MULTIVARIABLE CRYPTOSYSTEMS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — Bc. ALICE DOKTOROVÁ
AUTHOR

VEDOUCÍ PRÁCE — doc. RNDr. MIROSLAV KUREŠ, Ph.D.
SUPERVISOR

BRNO 2013

Vysoké učení technické v Brně, Fakulta strojního inženýrství

Ústav matematiky
Akademický rok: 2012/2013

ZADÁNÍ DIPLOMOVÉ PRÁCE

student(ka): Bc. Alice Doktorová

který/která studuje v **magisterském navazujícím studijním programu**

obor: **Matematické inženýrství (3901T021)**

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma diplomové práce:

Gröbnerovy báze, Čuang-c'ův algoritmus a ataky multivariačních kryptosystémů

v anglickém jazyce:

Gröbner basis, Zhuang-Zi algorithm and attacks of multivariable cryptosystems

Stručná charakteristika problematiky úkolu:

Jde o téma z aplikované komutativní algebry, a sice řešení soustav polynomiálních rovnic více neurčitých speciálními algoritmy. Lze použít při útocích na multivariační kryptosystémy.

Cíle diplomové práce:

(1) Kvalitní přehled teoretického aparátu z komutativní algebry se zaměření na Gröbnerovy báze

(2) Přehled multivariačních kryptosystémů (Matsumoto-Imai, TTM, ...) a útoků na ně

(3) Diskuse těchto útoků a návrhy na vylepšení

Seznam odborné literatury:

M. Roczen, First steps with Gröbner Bases, Lecture Notes

J. Ding, J. E. Gover, D. S. Schmidt, Zhuang-Zi: A New Algorithm for Solving Multivariate Polynomial Equations over a Finite Field, IACR

T. Sauer, Gröbner bases, H bases and interpolation, Trans. of Am. Math. Soc.

Vedoucí diplomové práce: doc. RNDr. Miroslav Kureš, Ph.D.

Termín odevzdání diplomové práce je stanoven časovým plánem akademického roku 2012/2013.

V Brně, dne

L.S.

prof. RNDr. Josef Šlapal, CSc. Ředitel ústavu	prof. RNDr. Miroslav Doupovec, CSc., dr. h. c. Děkan fakulty

Abstrakt
Tato diplomová práce je zaměřena na multivariační kryptosystémy. Její součástí je přehled komutativní algebry se zaměřením na Gröbnerovy báze. Z algoritmů jsou studovány především ty, které využívají Gröbnerovy báze a to Buchbergerův algoritmus, který je již implementován v programu Wolfram Mathematica, a F4 algoritmus, pro který byl vytvořen programový balík v prostředí Wolfram Mathematica. Jako poslední je popsán Čuang-c'ův algoritmus, pro který byl pro zjednodušení vytvořen program pro počítání Lagrangeova interpolačního polynomu v jazyce Python.

Summary
This diploma thesis is devoted to the multivariate cryptosystems. It includes an overview of commutative algebra with emphasis on Gröbner bases. Of all algorithms, especially the ones using Gröbner bases are studied, i.e. Buchberger's algorithm, which is already implemented in Wolfram Mathematica, and F4 algorithm, for which a program package has been created in the Wolfram Mathematica environment. Also Zhuang-Zi algorithm is described. To simplify its steps a program to compute the Lagrange interpolation polynomial has been created in Python.

Klíčová slova
Multivariační interpolace, interpolační polynom, konečné pole, F4 algoritmus, Čuang-c'ův algoritmus, Buchbergerův algoritmus, multivariační kryptosystémy

Keywords
Multivariable interpolation, interpolation polynomial, finite field, F4 algorithm, Zhuang-Zi algorithm, Buchberger algorithm, multivariable cryptosystems

DOKTOROVÁ, A. *Gröbnerovy báze, Čuang-c'ův algoritmus a ataky multivariačních kryptosystémů*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2013. 76 s. Vedoucí doc. RNDr. Miroslav Kureš, Ph.D.

Prohlašuji, že jsem diplomovou práci „Gröbnerovy báze, Čuang-c'ův algoritmus a ataky multivariačních kryptosystémů“ vypracovala samostatně s použitím odborné literatury a pramenů, uvedených na seznamu, jenž je součástí této práce.

Alice Doktorová

Děkuji panu doc. RNDr. Miroslavu Kurešovi, Ph.D. za rady, věnovaný čas a odborné vedení při tvorbě této diplomové práce.

Alice Doktorová

Obsah

1. Úvod

Cílem diplomové práce bylo sepsat přehled komutativní algebry se zaměřením na Gröbnerovy báze. Dále prostudovat vybrané multivariační kryptosystémy a ataky na ně. Posledním úkolem bylo implementovat dané algoritmy.

Diplomová práce je rozdělena do tří kapitol. První kapitola dává přehled komutativní algebry se zaměřením na Gröbnerovy báze. Jsou zde studovány také ideály a eliminační teorie. Ve druhé kapitole jsou sepsány dané multivariační kryptosystémy, včetně obecného úvodu. V této práci jsou studovány algoritmy využívající Gröbnerovy báze, a to Buchbergerův algoritmus a F4 algoritmus. Z dalších algoritmů je zde popsán Čuang-c'ův algoritmus. V poslední kapitole se budeme zabývat implementací daných algoritmů. Pro Buchbergerův algoritmus budeme studovat ukázkové volání již implementovaného algoritmu v prostředí Wolfram Mathematica. Pro F4 algoritmus je naším úkolem sepsat programový balík v prostředí Wolfram Mathematica. Pro Čuang-c'ův algoritmus využijeme k implementaci programovací jazyk Python, kde vytvoříme program pro výpočet Lagrangeova interpolačního polynomu nad daným rozšířeným konečným polem.

Diplomová práce obsahuje také přílohu s příklady. Příklady jsou zaměřené na komutativní algebru a to na výpočet ideálů, Gröbnerových bází, S-polynomů a také implicitizace. Dále jsou přiloženy kódy pro F4 algoritmus a pro výpočet Lagrangeova interpolačního polynomu.

2. Komutativní algebra

Tato kapitola je inspirována [1].

2.1. Ideály v okruzích

2.1.1. Afinní variety

Definice 1. Nechť k je komutativní pole a $f_1, ..., f_s$ jsou polynomy v $k[x_1, ..., x_n]$. Pak

$$V(f_1, ..., f_s) = \{(a_1, ..., a_n) \in k^n; f_i(a_1, ..., a_n) = 0; \forall i : 1 \leq i \leq s\}$$

nazveme *afinní varietou* generovanou $f_1, ..., f_s$.

Tedy afinní varieta $V(f_1, ..., f_s) \subset k^n$ je množina všech řešení soustavy rovnic

$$f_1(x_1, ..., x_n) = ... = f_s(x_1, ..., x_n) = 0.$$

Lemma 2. *Jestliže $V, W \subset k^n$ jsou afinní variety, pak také $V \cup W$ a $V \cap W$ jsou afinní variety. Navíc platí*

$$V \cap W = V(f_1, ..., f_s, g_1, ..., g_t); V \cup W = V(f_i g_j), pro\ 1 \leq i \leq s, 1 \leq j \leq t.$$

2.1.2. Parametrizace afinních variet

Předpokládejme, že je dána varieta $V = V(f_1, ..., f_s) \subset k^n$. Pak racionální parametrická reprezentace V obsahuje racionální funkce $r_1, ..., r_n \in k(t_1, ..., t_m)$ takové, že body dané vztahy

$$\begin{aligned} x_1 &= r_1(t_1, ..., t_m), \\ &\vdots \\ x_n &= r_n(t_1, ..., t_m) \end{aligned}$$

leží ve V. Dále požadujeme, aby V byla nejmenší varieta obsahující tyto body.

V mnoha situacích máme parametrizaci variety V, kde $r_1, ..., r_n$ jsou polynomy. Takové případy nazýváme polynomická parametrická reprezentace V. Naproti tomu, původní rovnice $f_1 = ... = f_s = 0$ určující V, nazýváme implicitní reprezentace V.

2.1.3. Ideály

Definice 3. Množina $\mathfrak{i} \subset k[x_1, ..., x_n]$ je *ideál*, jestliže platí
i) $0 \in \mathfrak{i}$
ii) jestliže $f, g \in \mathfrak{i}$, pak také $f + g \in \mathfrak{i}$
iii) jestliže $f \in \mathfrak{i}$ a $h \in k[x_1, ..., x_n]$, pak $h \cdot f \in \mathfrak{i}$.

Definice 4. Nechť $f_1, ..., f_s$ jsou polynomy v $k[x_1, ..., x_n]$, pak množina

$$\langle f_1, ..., f_s \rangle = \left\{ \sum_{i=1}^{s} h_i f_i : h_1, ..., h_s \in k[x_1, ..., x_n] \right\}$$

je ideál.

Lemma 5. *Jestliže* $f_1, ..., f_s \in k[x_1, ..., x_n]$, *pak* $\langle f_1, ..., f_s \rangle$ *je ideál v* $k[x_1, ..., x_n]$. $\langle f_1, ..., f_s \rangle$ *nazveme ideál generovaný* $f_1, ..., f_s$.

Důkaz. 1. $0 \in \langle f_1, ..., f_s \rangle$, protože $0 = \sum_{i=1}^{s} 0 f_i$
2. Nechť $f = \sum_{i=1}^{s} p_i f_i, g = \sum_{i=1}^{s} q_i f_i, h \in k[x_1, ..., x_n]$. Pak dostáváme
$f + g = \sum_{i=1}^{s} (p_i + q_i) f_i$
$hf = \sum_{i=1}^{s} (hp_i) f_i$.
Dokázali jsme tedy, že $\langle f_1, ..., f_s \rangle$ je ideál. □

Ideál $\mathfrak{i} \subset k[x_1, ..., x_n]$ je konečně generovaný, pokud existují $f_1, ..., f_s \in k[x_1, ..., x_n]$ tekové, že $\mathfrak{i} = \langle f_1, ..., f_s \rangle$ a $f_1, ..., f_s$ tvoří bázi ideálu $\mathfrak{i}$.

Tvrzení 6. *Jestliže* $f_1, ..., f_s$ *a* $g_1, ..., g_t$ *jsou báze stejného ideálu v* $k[x_1, ..., x_n]$ *takové, že* $\langle f_1, ..., f_s \rangle = \langle g_1, ..., g_t \rangle$, *pak* $V(f_1, ..., f_s) = V(g_1, ..., g_t)$.

Příklad: Uvažujme varietu $V(2x^2+3y^2-11, x^2-y^2-3)$. Snadno se ukáže, že $2x^2+3y^2-11 = 2(x^2-4)+3(y^2-1); x^2-y^2-3 = 1(x^2-4)-1(y^2-1)$. Tedy $\langle 2x^2+3y^2-11, x^2-y^2-3 \rangle = \langle x^2-4, y^2-1 \rangle$ tak, že $V(2x^2+3y^2-11, x^2-y^2-3) = V(x^2-4, y^2-1) = \{\pm 2, \pm 1\}$.

Definice 7. Nechť $V \subset k^n$ je afinní varieta. Pak množina

$$\mathfrak{i}(V) = \{f \in k[x_1, ..., x_n] : f(a_1, ..., a_n) = 0, \forall (a_1, ..., a_n) \in V\}.$$

Zásadním výsledkem je, že $\mathfrak{i}(V)$ je ideál.

Lemma 8. *Jestliže* $V \subset k^n$ *je afinní varieta, pak* $\mathfrak{i}(V) \subset k[x_1, ..., x_n]$ *je ideál.*

Důkaz. Je zřejmé, že $0 \in \mathfrak{i}(V)$, protože nulový polynom je nulový pro všechna k^n. Dále předpokládejme, že $f, g \in \mathfrak{i}(V)$ a $h \in k[x_1, ..., x_n]$. Nechť $(a_1, ..., a_n)$ je libovolný bod V. Pak $f(a_1, ..., a_n) + g(a_1, ..., a_n) = 0 + 0 = 0$
$h(a_1, ..., a_n) f(a_1, ..., a_n) = h(a_1, ..., a_n) 0 = 0$. Tím je dokázáno, že $\mathfrak{i}(V)$ je ideál. □

Definice 9. $\mathfrak{i}(V)$ z předchozího Lemmatu budeme nazývat ideál variety V.

Lemma 10. *Jestliže* $f_1, ..., f_s \in k[x_1, ..., x_n]$, *pak* $\langle f_1, ..., f_s \rangle \subset \mathfrak{i}(V(f_1, ..., f_s))$. *Rovnost nastat nemusí.*

Tvrzení 11. *Nechť* V *a* W *jsou afinní variety v* k^n. *Pak*
i) $V \subset W$ *právě tehdy, když* $\mathfrak{i}(V) \supset \mathfrak{i}(W)$
ii) $V = W$ *právě tehdy, když* $\mathfrak{i}(V) = \mathfrak{i}(W)$.

2.1.4. Polynomy jedné proměnné

Definice 12. Mějme nenulový polynom $f \in k[x]$, nechť $f = a_0 x^m + a_1 x^{m-1} + \cdots + a_m$, kde $a_i \in k$ a $a_0 \neq 0$, tedy $deg(f) = m$. Pak řekneme, že $a_0 x^m$ je *hlavní člen* f, značíme $LT(f) = a_0 x^m$.

Tvrzení 13 (Algoritmus dělení). *Nechť* k *je pole a nechť* g *je nenulový polynom v* $k[x]$. *Pak každé* $f \in k[x]$ *můžeme psát* $f = qg + r$, *kde* $q, r \in k[x]$ *a* $r = 0$ *nebo* $deg(r) < deg(g)$. *Navíc* q *a* r *jsou jednoznačné.*

Tvrzení 14. *Jestliže k je pole a $f \in k[x]$ je nenulový polynom, pak f má nejvýše $deg(f)$ kořenů v k.*

Tvrzení 15. *Jestliže k je pole, pak každý ideál v $k[x]$ může být zapsán ve tvaru $\langle f \rangle$ pro nějaké $f \in k[x]$. Navíc f je jednoznačné vzhledem k násobení nenulovou konstantou v k.*

Ideál generovaný jedním prvkem se nazývá *hlavní* ideál.

Definice 16. *Největší společný dělitel* polynomů $f, g \in k[x]$ je množina polynomů h s vlastností
i) h dělí f a g
ii) Jestliže p je další polynom, který dělí f a g, pak p dělí h.
Má-li h tyto vlastnosti, pak píšeme $h = GCD(f, g)$.

Tvrzení 17. *Nechť $f, g \in k[x]$. Pak*
i) $GCD(f, g)$ existuje a je neprázdná a jsou-li dva polynomy jejími prvky, pak jeden je konstantním násobkem druhého
ii) $GCD(f, g)$ je generátor ideálu $\langle f, g \rangle$
iii) existuje algoritmus pro nalezení $GCD(f, g)$ (Euklidův algoritmus).

Příklad: Spočtěte $GCD(x^4 - 1, x^6 - 1)$. Nejprve použijeme algoritmus dělení
$x^4 - 1 = 0(x^6 - 1) + x^4 - 1$
$x^6 - 1 = x^2(x^4 - 1) + x^2 - 1$
$x^4 - 1 = (x^2 - 1)(x^2 + 1) + 0$
$GCD(x^4 - 1, x^6 - 1) = GCD(x^4 - 1, x^2 - 1) = GCD(x^2 - 1, 0) = x^2 - 1$. Poznamenejme, že takto spočítaný GCD je odpovědí na nalezení generátoru ideálu $\langle x^4 - 1, x^6 - 1 \rangle = \langle x^2 - 1 \rangle$.

Definice 18. *Největší společný dělitel* polynomů $f_1, ..., f_s \in k[x]$ je množina polynomů h s vlastností
i) h dělí $f_1, ..., f_s$
ii) jestliže p je další polynom, který dělí $f_1, ..., f_s$, pak p dělí h, a značíme $h = GCD(f_1, ..., f_s)$.

Tvrzení 19. *Nechť $f_1, ..., f_s \in k[x]$, kde $s \geq 2$, pak*
i) $GCD(f_1, ..., f_s)$ existuje a je jednoznačný vzhledem k násobení nenulovou konstantou v k
ii) $GCD(f_1, ..., f_s)$ je generátor ideálu $\langle f_1, ..., f_s \rangle$
iii) jestliže $s \geq 3$, pak $GCD(f_1, ..., f_s) = GCD(f_1, GCD(f_2, ..., f_s))$
iv) existuje algoritmus pro nalezení $GCD(f_1, ..., f_s)$.

Příklad: Uvažujme ideál $\langle x^3 - 3x + 2, x^4 - 1, x^6 - 1 \rangle \subset k[x]$. Víme, že $GCD(x^3 - 3x + 2, x^4 - 1, x^6 - 1)$ je generátor. Navíc můžeme ověřit, že $GCD(x^3 - 3x + 2, x^4 - 1, x^6 - 1) = GCD(x^3 - 3x + 2, GCD(x^4 - 1, x^6 - 1)) = GCD(x^3 - 3x + 2, x^2 - 1) = x - 1$. Dostáváme $\langle x^3 - 3x + 2, x^4 - 1, x^6 - 1 \rangle = \langle x - 1 \rangle$.

2.2. Gröbnerovy báze

2.2.1. Monomické uspořádání

Definice 20. *Monomické uspořádání* v $k[x_1, ..., x_n]$ je libovolná relace $>$ na $\mathbb{N}_0^n$, nebo ekvivalentně, každá relace na množině monomů $x^\alpha, \alpha \in \mathbb{N}_0^n$ splňující

i) $>$ je úplné (nebo lineární) uspořádání na $\mathbb{N}_0^n$
ii) jestliže $\alpha > \beta$ a $\gamma \in \mathbb{N}_0^n$, pak $\alpha + \gamma > \beta + \gamma$
iii) $>$ je dobré uspořádání na $\mathbb{N}_0^n$. To znamená, že každá neprázdná podmnožina $\mathbb{N}_0^n$ má nejmenší prvek.

Lemma 21. *Relace uspořádání $>$ na $\mathbb{N}_0^n$ je dobré uspořádání právě tehdy, když každá klesající posloupnost v $\mathbb{N}_0^n$ je konečná.*

Definice 22 (**Lexikografické uspořádání**)**.** Nechť $\alpha = (\alpha_1, ..., \alpha_n), \beta = (\beta_1, ..., \beta_n) \in \mathbb{N}_0^n$. Řekneme, že $\alpha >_{lex} \beta$, jestliže vektorový rozdíl $\alpha - \beta \in \mathbb{Z}^n$ má první nenulovou složku kladnou. Píšeme $x^\alpha >_{lex} x^\beta$, jestliže $\alpha >_{lex} \beta$.

Příklad:
a)$(3,2,3) >_{lex} (1,3,6)$, protože $\alpha - \beta = (2,-1,-3)$
b)$(1,4,3) >_{lex} (1,4,2)$, protože $\alpha - \beta = (0,0,1)$.

Tvrzení 23. *Lexikografické uspořádání na $\mathbb{N}_0^n$ je monomické uspořádání.*

Definice 24 (**Stupňované lexikografické uspořádání**)**.** Nechť $\alpha, \beta \in \mathbb{N}_0^n$. Řekneme, že $\alpha >_{grlex} \beta$, jestliže $|\alpha| = \sum_{i=1}^n \alpha_i > |\beta| = \sum_{i=1}^n \beta_i$ nebo $|\alpha| = |\beta|$ a zároveň $\alpha >_{lex} \beta$.

Při stupňovaném lexikografickém uspořádání nejdříve uvažujeme celkový stupeň monomu.

Příklad:
a)$(1,2,3) >_{grlex} (3,2,0)$, protože $|(1,2,3)| = 6 > |(3,2,0)| = 5$
b)$(1,2,4) >_{grlex} (1,1,5)$, protože $|(1,2,4)| = |(1,1,5)|$ a zároveň $(1,2,4) >_{lex} (1,1,5)$.

Definice 25 (**Stupňované inverzní lexikografické uspořádání**)**.** Nechť $\alpha, \beta \in \mathbb{N}_0^n$. Řekneme, že $\alpha >_{grevlex} \beta$, jestliže $|\alpha| = \sum_{i=1}^n \alpha_i > |\beta| = \sum_{i=1}^n \beta_i$ nebo $|\alpha| = |\beta|$ a poslední nenulová složka $\alpha - \beta \in \mathbb{Z}^n$ je záporná.

Příklad:
a)$(4,7,1) >_{grevlex} (4,2,3)$, protože $|(4,7,1)| = 12 > |(4,2,3)| = 9$
b)$(1,5,2) >_{grevlex} (4,1,3)$, protože $|(1,5,2)| = |(4,1,3)|$ a zároveň $(1,5,2) - (4,1,3) = (-3,4,-1)$.

Rozdíl mezi stupňovaným lexikografickým a stupňovaným inverzním lexikografickým uspořádáním nastává, pokud je celkový stupeň monomů shodný. Stupňované lexikografické uspořádání dále uplatňuje lexikografické uspořádání a dává přednost vyšší mocnině u největší proměnné. Stupňované inverzní lexikografické uspořádání dává přednost nižší mocnině u nejmenší proměnné.

Příklad: Uvažujme polynom $f = 5xy^2z^2 + 6z - 11x^3 + 9x^2z^3 \in k[x,y,z]$
a) lexikografické uspořádání: $f = -11x^3 + 9x^2z^3 + 5xy^2z^2 + 6z$
b) stupňované lexikografické uspořádání: $f = 9x^2z^3 + 5xy^2z^2 - 11x^3 + 6z$
c) stupňované inverzní lexikografické uspořádání: $f = 5xy^2z^2 + 9x^2z^3 - 11x^3 + 6z$.

Definice 26. Nechť $f = \sum_\alpha a_\alpha x^\alpha$ je nenulový polynom v $k[x_1, ..., x_n]$ a nechť $>$ je monomické uspořádání
i) *multistupeň* polynomu f definujeme jako $multideg(f) = max(\alpha \in \mathbb{N}_0^n; a_\alpha \neq 0)$
ii) *hlavní koeficient* polynomu f definujeme jako $LC(f) = a_{multideg(f)} \in k$
iii) *hlavní monom* polynomu f definujeme jako $LM(f) = x^{multideg(f)}$

Poznámka: Hlavní člen polynomu f vyjádříme jako $LT(f) = LC(f)LM(f)$.

Příklad: Je dán polynom $f = 5xy^2z^2 + 6z - 11x^3 + 9x^2z^3$. Uvažujme lexikografické uspořádání. Poté $multideg(f) = (3,0,0), LC(f) = -11, LM(f) = x^3, LT(f) = -11x^3$.

Lemma 27. *Nechť $f, g \in k[x_1, ..., x_n]$ jsou nenulové polynomy. Pak*
i) $multideg(fg) = multideg(f) + multideg(g)$
ii) Jestliže $f + g \neq 0$,pak $multideg(f + g) \leq max(multideg(f), multideg(g))$. Když navíc platí, že $multideg(f) \neq multideg(g)$, pak nastává rovnost.

2.2.2. Algoritmus dělení v $k[x_1, ..., x_n]$

Příklad: Je dán polynom $f = x^2y+xy^2+y^2$. Vydělte jej polynomy $f_1 = xy-1, f_2 = y^2-1$ s použitím lexikografického uspořádání s $x > y$.

$$x^2y + xy^2 + y^2 : \begin{cases} xy - 1 \\ y^2 - 1 \end{cases} = x + y$$

$$\begin{array}{l} \underline{x^2y - x} \\ xy^2 + x + y^2 \\ \underline{xy^2 - y} \\ x + y^2 + y \end{array}$$

Zbytek po dělení je $x + y^2 + y$. Ani $LT(f_1)$ ani $LT(f_2)$ nedělí $LT(x + y^2 + y)$, ale $x+y^2+y$ není zbytek, protože $LT(f_2)$ dělí y^2. Tudíž x přesuneme do zbytku a pokračujeme dále v dělení. Obecně můžeme říci, že pokud nemůžeme dělit ani $LT(f_1)$ ani $LT(f_2)$, pak přesuneme hlavní člen do zbytku a v dělení pokračujeme dále. Po dokončení dělení v našem příkladu dostaneme výsledek $x^2y + xy^2 + y^2 = (x + y)(xy - 1) + 1(y^2 - 1) + x + y + 1$.

Věta 28 (Algoritmus dělení v $k[x_1, ..., x_n]$). *Mějme monomické uspořádání $>$ na $\mathbb{N}_0^n$ a nechť $F = (f_1, ..., f_s)$ je s-tice polynomů v $k[x_1, ..., x_n]$. Pak každé $f \in k[x_1, ..., x_n]$ můžeme zapsat jako*

$$f = a_1f_1 + \cdots + a_sf_s + r,$$

kde $a_i, r \in k[x_1, ..., x_n]$ a $r = 0$ nebo r je lineární kombinace monomů s koeficienty v k, kde žádný z nich není dělitelný žádným $LT(f_1), ..., LT(f_s)$. r nazveme zbytek po dělení F. Navíc jestliže $a_if_i \neq 0$, pak platí $multideg(f) \geq multideg(a_if_i)$.

Příklad: Je dán polynom $x^2y+xy^2+y^2$. Vydělte jej polynomy $f_1 = y^2-1, f_2 = xy-1$.

$$x^2y + xy^2 + y^2 : \begin{cases} y^2 - 1 \\ xy - 1 \end{cases} = \begin{matrix} x + 1 \\ x \end{matrix}$$

$$\begin{array}{lll} \underline{x^2y - x} & & \\ xy^2 + x + y^2 & & \\ \underline{xy^2 - x} & & \\ \underline{2x + y^2} & \rightarrow & 2x \\ y^2 & & \\ \underline{y^2 - 1} & & \\ \underline{1} & \rightarrow & 2x + 1 \\ 0 & & \end{array}$$

Výsledkem tedy je $x^2y + xy^2 + y^2 = (x + 1)(y^2 - 1) + x(xy - 1) + 2x + 1$.
Je tedy zřejmé, že při záměně dělitelů je zbytek po dělení jiný.

Algoritmus dělení polynomů více proměnných souvisí s řešením problému příslušnosti k ideálu. Jestliže je zbytek f po dělení $F = (f_1, ..., f_s)$ nulový, pak $f \in \langle f_1, ..., f_s \rangle$. Vidíme tedy, že $r = 0$ je postačující podmínka pro příslušnost k ideálu, ale není to podmínka nutná (viz následující příklad).

Příklad: Nechť $f_1 = xy + 1, f_2 = y^2 - 1 \in k[x]$ s lexikografickým uspořádáním.
Vydělte polynom $f = xy^2 - x$ dvojicí polynomů $F = (f_1, f_2)$. Obdržíme řešení tvaru $xy^2 - x = y(xy + 1) + 0(y^2 - 1) + (-x - y)$.
Dělení dvojicí $F = (f_2, f_1)$ vede k výsledku $xy^2 - x = x(y^2 - 1) + 0(xy + 1) + 0$.
Odtud vidíme, že $f \in \langle f_1, f_2 \rangle$ a přesto při dělení $F = (f_1, f_2)$ je zbytek nenulový.

2.2.3. Monomické ideály a Dicksonova věta

Definice 29. Ideál $\mathfrak{i} \subset k[x_1, ..., x_n]$ nazveme *monomický ideál*, jestliže existuje podmnožina $A \subset \mathbb{N}_0^n$ (i nekonečná) taková, že $\mathfrak{i}$ obsahuje všechny polynomy ve tvaru konečného součtu $\sum_{\alpha \in A} h_\alpha x^\alpha$, kde $h_\alpha \in k[x_1, ..., x_n]$. V tomto případě píšeme $\mathfrak{i} = \langle x^\alpha; \alpha \in A \rangle$.

Příklad monomického ideálu je ideál $\mathfrak{i} = \langle xy, x^2y^3, x^3y \rangle \subset k[x, y]$. Příklad ideálu, který není monomický je ideál $\mathfrak{i}_1 = \langle xy - y, x^2y + xy^2 \rangle$.

Lemma 30. *Nechť $\mathfrak{i} = \langle x^\alpha; \alpha \in A \rangle$ je monomický ideál. Pak monom x^β leží v $\mathfrak{i}$ právě tehdy, když x^β je dělitelné x^α pro nějaké $\alpha \in A$.*

Lemma 31. *Nechť $\mathfrak{i}$ je monomický ideál a nechť $f \in k[x_1, ..., x_n]$. Pak jsou následující tvrzení ekvivalentní.*
i) $f \in \mathfrak{i}$
ii) každý člen f leží v $\mathfrak{i}$
iii) f je lineární kombinací monomů v $\mathfrak{i}$ s koeficienty z k.

Tvrzení 32. *Dva monomické ideály jsou totožné právě tehdy, když obsahují stejné monomy.*

Každý monomický ideál z $k[x_1, ..., x_n]$ je konečně generovaný.

Věta 33 (Dicksonova věta). *Nechť $\mathfrak{i} = \langle x^\alpha; \alpha \in A \rangle \subseteq k[x_1, ..., x_n]$ je monomický ideál. Pak můžeme $\mathfrak{i}$ psát ve tvaru $\mathfrak{i} = \left\langle x^{\alpha(1)}, ..., x^{\alpha(s)} \right\rangle$, kde $\alpha(1), ..., \alpha(s) \in A$. Ideál $\mathfrak{i}$ má tedy konečnou bázi.*

Tvrzení 34. *Nechť $>$ je relace na $\mathbb{N}_0^n$ splňující*
i) $>$ je úplné uspořádání na $\mathbb{N}_0^n$
ii) jestliže $\alpha > \beta$ a $\gamma \in \mathbb{N}_0^n$, pak $\alpha + \gamma > \beta + \gamma$.
Pak $>$ je dobré uspořádání právě tehdy, když $\alpha \geq 0$ pro $\forall \alpha \in \mathbb{N}_0^n$.

2.2.4. Věta o Hilbertově bázi a Gröbnerovy báze

Definice 35. Nechť $\mathfrak{i} \subset k[x_1, ..., x_n]$ je ideál různý od $\{0\}$ (tzn. obsahuje alespoň jeden nenulový polynom)
i) Označme $LT(\mathfrak{i})$ množinu hlavních členů prvků z $\mathfrak{i}$. Tedy
$LT(\mathfrak{i}) = \{cx^\alpha; ex. f \in \mathfrak{i}, \text{pro které } LT(f) = cx^\alpha\}$
ii) Označme $\langle LT(\mathfrak{i}) \rangle$ ideál generovaný prvky $LT(\mathfrak{i})$.

Příklad: Nechť $\mathfrak{i} = \langle f_1, f_2 \rangle, f_1 = x^3 - 2xy, f_2 = x^2y - 2y^2 + x$, použijeme stupňované lexikografické uspořádání monomů v $k[x, y]$.Potom $x(x^2y - 2y^2 + x) - y(x^3 - 2xy) = x^2$ a tudíž $x^2 \in \mathfrak{i}$. $LT(x^2) \in \langle LT(\mathfrak{i}) \rangle$, ale x^2 není dělitelné ani $LT(f_1)$ ani $LT(f_2)$ a tudíž $x^2 \in \langle LT(f_1), LT(f_2) \rangle$.

Tvrzení 36. *Nechť $\mathfrak{i} \subset k[x_1, ..., x_n]$ je ideál.*
i) $\langle LT(\mathfrak{i}) \rangle$ je monomický ideál
ii) Existují $g_1, ..., g_t \in \mathfrak{i}$ takové, že $\langle LT(\mathfrak{i}) \rangle = \langle LT(g_1), ..., LT(g_t) \rangle$.

Věta 37 (Hilbertova věta o bázi). *Každý ideál $\mathfrak{i} \subset k[x_1, ..., x_n]$ má konečnou generující množinu. Proto $\mathfrak{i} = \langle g_1, ..., g_t \rangle$ pro nějaké $g_1, ..., g_t \in \mathfrak{i}$.*

Důkaz. Pokud $\mathfrak{i} = \{0\}$, pak za generující množinu vezmeme $\{0\}$, která je určitě konečná. Jestliže $\mathfrak{i}$ obsahuje nějaký nenulový polynom, pak dle předešlé věty a dle Dicksonovy věty existují $g_1, ..., g_t$ takové, že $\langle LT(\mathfrak{i}) \rangle = \langle LT(g_1), ..., LT(g_t) \rangle$. Předpokládejme, že $\mathfrak{i} = \langle g_1, ..., g_t \rangle$. Je zřejmé, že $\langle g_1, ..., g_t \rangle \subset \mathfrak{i}$, jelikož každé $g_i \in \mathfrak{i}$. Vezmeme libovolný polynom $f \in \mathfrak{i}$ a vydělíme jej polynomy $g_1, ..., g_t$. Poté můžeme psát $f = a_1g_1 + \cdots + a_tg_t + r$, kde žádný člen r není dělitelný $LT(g_1), ..., LT(g_t)$. Vidíme, že $r = f - a_1g_1 - \cdots - a_tg_t \in \mathfrak{i}$. Pokud je $r \neq 0$, pak nutně $LT(r) \in \langle LT(\mathfrak{i}) \rangle = \langle LT(g_1), ..., LT(g_t) \rangle$, protože je $\langle LT(\mathfrak{i}) \rangle$ monomický, musí být $LT(r)$ dělitelný některým z generátorů $LT(g_i)$. Tím dostáváme spor s tím, že r je zbytek po dělení. Proto musí být $r = 0$. Pak $f = a_1g_1 + \cdots + a_tg_t \in \langle g_1, ..., g_t \rangle$. Odtud plyne $\mathfrak{i} \subset \langle g_1, ..., g_t \rangle$ a důkaz je hotov. □

Definice 38. Mějme monomické uspořádání. Konečnou podmnožinu $G = \{g_1, ..., g_t\}$ ideálu $\mathfrak{i}$ nazveme *Gröbnerova báze* (nebo standartní báze), jestliže $\langle LT(g_1), ..., LT(g_t) \rangle = \langle LT(\mathfrak{i}) \rangle$.

Ekvivalentně můžeme říci, že $\{g_1, ..., g_t\} \subset \mathfrak{i}$ je Gröbnerova báze právě tehdy, když hlavní člen libovolného prvku $\mathfrak{i}$ je dělitelný $LT(g_i)$ pro nějaké i.

Tvrzení 39. *Mějme monomické uspořádání. Pak každý ideál $\mathfrak{i} \subset k[x_1, ..., x_n]$ různý od $\{0\}$ má Gröbnerovu bázi. Navíc každá množina polynomů $g_1, ..., g_t \in \mathfrak{i}$, pro kterou platí $\langle LT(\mathfrak{i}) \rangle = \langle LT(g_1), ..., LT(g_t) \rangle$, je Gröbnerova báze ideálu $\mathfrak{i}$.*

Tvrzení 40 (Podmínka vzestupné řady). *Nechť $\mathfrak{i}_1 \subset \mathfrak{i}_2 \subset \mathfrak{i}_3 \subset ...$ je vzestupná řada ideálů v $k[x_1, ..., x_n]$. Pak existuje $N \geq 1$ tak, že $\mathfrak{i}_N = \mathfrak{i}_{N+1} = \mathfrak{i}_{N+2} = ...$*

Definice 41. Nechť $\mathfrak{i} \subset k[x_1, ..., x_n]$ je ideál. Označíme $V(\mathfrak{i})$ množinu

$$V(\mathfrak{i}) = \{(a_1, ..., a_n) \in k^n; f(a_1, ..., a_n) = 0; \forall f \in \mathfrak{i}\}.$$

Tvrzení 42. *$V(\mathfrak{i})$ je afinní varieta. Tedy jestliže $\mathfrak{i} = \langle f_1, ..., f_s \rangle$, pak $V(\mathfrak{i}) = V(f_1, ..., f_s)$.*

2.2.5. Vlastnosti Gröbnerových bází

Tvrzení 43. *Nechť $G = \{g_1, ..., g_t\}$ je Gröbnerova báze ideálu $\mathfrak{i} \subset k[x_1, ..., x_n]$ a nechť $f \in k[x_1, ..., x_n]$. Pak existuje jediné $r \in k[x_1, ..., x_n]$ s následujícími vlastnostmi*
i) žádný člen r není dělitelný žádným $LT(g_1), ..., LT(g_t)$
ii) existuje $g \in \mathfrak{i}$ takové, že $f = g + r$.

Tvrzení 44. *Nechť $G = \{g_1, ..., g_t\}$ je Gröbnerova báze ideálu $\mathfrak{i} \subset k[x_1, ..., x_n]$ a nechť $f \in k[x_1, ..., x_n]$. Pak $f \in \mathfrak{i}$ právě tehdy, když zbytek po dělení f množinou G je nulový.*

Toto tvrzení je někdy použito jako definice Gröbnerovy báze, protože je ekvivalentní s podmínkou $\langle LT(g_1), ..., LT(g_t)\rangle = \langle LT(\mathfrak{i})\rangle$.

Definice 45. Označme $\overline{f}^F$ zbytek po dělení f uspořádánou s-ticí $F = (f_1, ..., f_s)$. Je-li F Gröbnerova báze pro $(f_1, ..., f_s)$, pak se na F můžeme dívat jako na množinu (bez uspořádání).

Příklad: Mějme polynom x^5y. Je dána $F = (x^2y - y^2, x^4y^2 - y^2) \subset k[x, y]$. Použijeme lexikografické uspořádání.
Podle algebraického dělení dostaneme $x^5y = (x^3 + xy)(x^2y - y^2) + 0(x^4y^2 - y^2) + xy^3$ a píšeme $\overline{x^5y}^F = xy^3$.

Definice 46. Nechť $f, g \in k[x_1, ..., x_n]$ jsou nenulové polynomy.
i) Jestliže $multideg(f) = \alpha$ a $multideg(g) = \beta$, pak zaveďme $\gamma = (\gamma_1, ..., \gamma_n)$, kde $\gamma_i = max(\alpha_i\beta_i); \forall i$. x^γ nazveme *nejmenší společný násobek* $LM(f), LM(g)$, píšeme $x^\gamma = LCM(LM(f), LM(g))$.
ii) Zaveďme *S-polynom* f, g předpisem $S(f, g) = \frac{x^\gamma}{LT(f)}f - \frac{x^\gamma}{LT(g)}g$.

Příklad: Jsou dány polynomy $f = x^3y^2 - x^2y^3 + x, g = 3x^4y + y^2 \in \mathbb{R}[x, y]$, uvažujme stupňované lexikografické uspořádání. Potom $\gamma = (4, 2)$ a
$S(f, g) = \frac{x^4y^2}{x^3y^2}(x^3y^2 - x^2y^3 + x) - \frac{x^4y^2}{3x^4y}(3x^4y + y^2) = -x^3y^3 + x^2 - \frac{1}{3}y^3$.

Lemma 47. *Předpokládejme, že máme součet $\sum_{i=1}^s c_if_i$, kde $c_i \in k$ a $multideg(f_i) = \delta \in \mathbb{N}_0^n, \forall i$. Jestliže $multideg(\sum_{i=1}^s c_if_i) < \delta$, pak $\sum_{i=1}^s c_if_i$ je lineární kombinace S- polynomů $S(f_j, f_k)$ pro $1 \le j, k \ge s$ s koeficienty v k. Navíc každý $S(f_i, f_k)$ má maximální stupeň menší než δ.*

Věta 48 (**Buchbergerovo kritérium**). *Nechť $\mathfrak{i}$ je polynomický ideál. Pak báze $G = \{g_1, ..., g_t\}$ je Gröbnerova báze $\mathfrak{i}$ právě tehdy, když pro všechny dvojice $i, j; i \neq j$ je zbytek po dělení $S(g_i, g_j)$ bází G nulový.*

Příklad: Mějme ideál $\mathfrak{i} = \langle y - x^2, z - x^3\rangle$ a ukažme, že $G = \{y - x^2, z - x^3\}$ je Gröbnerova báze vzhledem k lexikografickému uspořádání pro $y > z > x$.
Báze G má pouze dva členy, tudíž stačí ověřit, že zbytek po dělení S- polynomu prvky báze G je nulový. $S(y - x^2, z - x^3) = \frac{yz}{y}(y - x^2) - \frac{yz}{z}(z - x^3) = yx^3 - zx^2$. Provedeme algebraické dělení a dostaneme $yx^3 - zx^2 = x^3(y - x^2) + (-x^2)(z - x^3) + 0$ a tedy $\overline{S(y - x^2, z - x^3)}^G = 0$. Zjistili jsme tedy, že G je Gröbnerova báze.

2.2.6. Aplikace Gröbnerových bází

Problém příslušnosti k ideálu

Jestliže zkombinujeme Gröbnerovy báze s algoritmem dělení, dostaneme následující algoritmus příslušnosti k ideálu: je dán ideál $\mathfrak{i} = \langle f_1, ..., f_s\rangle$. Můžeme rozhodnout, zda daný polynom f leží v ideálu $\mathfrak{i}$ následovně. Nejdříve použijeme algoritmus podobný Buchbergerovu algoritmu a najdeme Gröbnerovu bázi $G = \{g_1, ..., g_t\}$ ideálu $\mathfrak{i}$. Poté tvrzení 44 nám dává $f \in \mathfrak{i}$ právě tehdy, když $\overline{f}^G = 0$.

Příklad: Nechť $\mathfrak{i} = \langle f_1, f_2\rangle = \langle xz - y^2, x^3 - y^2\rangle \in \mathbb{C}[x, y, z]$. Použijeme stupňované lexikografické uspořádání. Nechť $f = -4x^2y^2z^2 + y^6 + 3z^5$. Platí, že $f \in \mathfrak{i}$?

Daná generující množina není Gröbnerova báze $\mathfrak{i}$, protože $LT(\mathfrak{i})$ obsahuje polynomy takové, že $LT(S(f_1, f_2)) = LT(-x^2y^2 + z^3) = -x^2y^2$ nejsou obsaženy v ideálu $\langle LT(f_1), LT(f_2)\rangle = \langle xz, x^3\rangle$. Tedy musíme spočítat Gröbnerovu bázi ideálu $\mathfrak{i}$.
$G = (f_1, ..., f_5) = (xz - y^2, x^3 - z^2, x^2y^2 - z^3, xy^4 - z^4, y^6 - z^5)$.

Poznamenejme, že máme redukovanou Gröbnerovu bázi. Nyní můžeme testovat, zda polynom patří do $\mathfrak{i}$. Například dělením f bází G dostáváme $f = (-4xy^2z - 4y^4)f_1 + 0f_2 + 0f_3 + 0f_4 + (-3)f_5 + 0$. Zbytek po dělení je nulový, tedy $f \in \mathfrak{i}$.

Problém řešení polynomických rovnic

Dále budeme vyšetřovat, jak můžeme použít Gröbnerovu bázi k řešení soustavy polynomických rovnic ve více proměnných.

Příklad: Uvažujme soustavu rovnic

$$\begin{aligned} x^2 + y^2 + z^2 &= 1 \\ x^2 + z^2 &= y \\ x &= z \end{aligned}$$

v $\mathbb{C}^3$. Tyto rovnice udávají $\mathfrak{i} = \langle x^2 + y^2 + z^2 - 1, x^2 + z^2 - y, x - z\rangle \subset \mathbb{C}[x, y, z]$. Cílem je najít všechny body ležící ve $V(\mathfrak{i})$. Spočítáme $V(\mathfrak{i})$ užitím libovolné báze $\mathfrak{i}$. Zkusme použít Gröbnerovu bázi. Použijeme Lexikografické uspořádání a dostaneme
$G = \left\{x - z, -y + 2z^2, z^4 + \frac{1}{2}z^2 - \frac{1}{4}\right\}$. Polynom g_3 závisí pouze na z a tudíž nejdříve spočítáme jeho kořeny. $z = \pm\frac{1}{2}\sqrt{\pm\sqrt{5} - 1}$, tedy máme 4 hodnoty pro z. Dosazením do zbývajících rovnic $g_1 = g_2 = 0$, získáme řešení pro y, x. Tím jsme našli všechna řešení pro zadaný systém rovnic.

Příklad: Najděte minimální a maximální hodnoty $x^3 + 2xyz - z^2$ vzhledem k omezení $x^2 + y^2 + z^2 = 1$. K řešení použijeme Lagrangeovy multiplikátory

$$\begin{aligned} 3x^2 + 2yz - 2x\lambda &= 0 \\ 2xz - 2y\lambda &= 0 \\ 2x^2 - 2z - 2z\lambda &= 0 \\ x^2 + y^2 + z^2 - 1 &= 0 \end{aligned}$$

Vypočítáme Gröbnerovu bázi pro ideál v $\mathbb{R}[x, y, z, \lambda]$ generovaný levými stranami rovnic. Použijeme lexikografické uspořádání $\lambda > x > y > z$. Gröbnerova báze je velice složitá, ale poslední polynom závisí pouze na proměnné z. Ostatní proměnné tedy můžeme vyeliminovat v procesu hledání Gröbnerovy báze. Rovnice obdržené kladením polynomů rovných nule dostáváme kořeny $z = 0, \pm 1, \pm\frac{2}{3}, \pm\frac{\sqrt{11}}{8\sqrt{2}}$.

Jestliže položíme z rovno každé z těchto hodnot, můžeme řešit zbytek rovnic pro x, y. Dostaneme následující řešení
$z = 0, y = 0, x = \pm 1$

$z = 0, y = \pm 1, x = 0$
$z = \pm 1, y = 0, x = 0$
$z = \frac{2}{3}, y = \frac{1}{3}, x = -\frac{2}{3}$
$z = -\frac{2}{3}, y = -\frac{1}{3}, x = -\frac{2}{3}$
$z = \frac{\sqrt{11}}{8\sqrt{2}}, y = -\frac{3\sqrt{11}}{8\sqrt{2}}, x = -\frac{3}{8}$
$z = -\frac{\sqrt{11}}{8\sqrt{2}}, y = \frac{3\sqrt{11}}{8\sqrt{2}}, x = -\frac{3}{8}$

Odtud je již snadné říci, která hodnota udává maximum a která minimum.

Problém implicitizace

Uvažujme, že parametrické rovnice

$$\begin{aligned} x_1 &= f_1(t_1, ..., t_m), \\ &\vdots \\ x_n &= f_n(t_1, ..., t_m) \end{aligned}$$

definují podmnožinu algebraické variety V v k^n. Např. to je vždy případ, kdy f_i jsou racionální funkce v proměnných $t_1, ..., t_m$.
Pro jednoduchost se omezíme pouze na případ, kdy f_i jsou polynomy. Mějme afinní varietu v k^{m+n} definovanou rovnicemi

$$\begin{aligned} x_1 - f_1(t_1, ..., t_m) &= 0, \\ &\vdots \\ x_n - f_n(t_1, ..., t_m) &= 0. \end{aligned}$$

Základní myšlenkou je eliminace proměnných $t_1, ..., t_m$ z těchto rovnic. Toto nám má dát rovnice pro V.

Použijeme lexikografické uspořádání v $k[t_1, ..., t_m, x_1, ..., x_n]$ definované uspořádáním proměnných $t_1 > ... > t_m > x_1 > ... > x_n$. Nyní předpokládejme, že máme Gröbnerovu bázi ideálu $\widetilde{\mathfrak{i}} = \langle x_1 - f_1, ..., x_n - f_n \rangle$. Protože používáme lexikografické uspořádání, očekáváme, že Gröbnerova báze má polynomy eliminující proměnné a $t_1, ..., t_m$ jsou eliminovány nejdříve, protože jsou největší v našem uspořádání. Tedy Gröbnerova báze $\widetilde{\mathfrak{i}}$ obsahuje polynomy, které obsahují pouze $x_1, ..., x_n$. Tedy tyto polynomy jsou kandidáty na rovnice V.

Příklad: Uvažujme parametrickou křivku $V : x = t^4, y = t^3, z = t^2$ v $\mathbb{C}^3$. Spočítejme Gröbnerovu bázi G ideálu $\mathfrak{i} = \langle t^4 - x, t^3 - y, t^2 - z \rangle$ s ohledem na lexikografické uspořádání v $\mathbb{C}[t, x, y, z]$ a najdeme $G = \{-t^2 + z, ty - z^2, tz - y, x - z^2, y^2 - z^3\}$. Poslední dva polynomy závisí pouze na x, y, z, tudíž definují afinní varietu $\mathbb{C}^3$ obsahující naši varietu V. Zbývá ověřit, že V je průnikem dvou prostorů $x - z^2 = 0, y^2 - z^3 = 0$.

2.3. Eliminační teorie

2.3.1. Věty o eliminaci a rozšíření

Abychom ukázali, jak eliminace funguje, podívejme se na následující příklad.

Příklad: Vyřešte systém rovnic

$$x^2 + y + z = 0$$
$$x + y^2 + z = 0$$
$$x + y + z^2 = 0$$

Jestliže ideál $\mathfrak{i} = \langle x^2 + y + z - 1, x + y^2 + z - 1, x + y + z^2 - 1\rangle$, pak Gröbnerova báze $\mathfrak{i}$ s ohledem na lexikografické uspořádání je dána polynomy $g_1 = x + y + z^2 - 1, g_2 = y^2 - y - z^2 + z, g_3 = 2yz^2 + z^4 - z^2, g_4 = z^6 - 4z^4 + 4z^3 - z^2$. Rovnice $g_4 = z^6 - 4z^4 + 4z^3 - z^2 = z^2(z-1)^2(z^2 + 2z - 1)$ obsahuje pouze z. Vidíme, že možné výsledky z jsou $0, 1, -1, \sqrt{2}, -\sqrt{2}$. Dosazením těchto hodnot do $g_2 = 0, g_3 = 0$ spočítáme možné hodnoty y a z rovnice $g_1 = 0$ dále vypočítáme x. Dostaneme, že soustava rovnic má právě pět řešení $(1,0,0), (0,1,0), (0,0,1), (-1+\sqrt{2}, -1+\sqrt{2}, -1+\sqrt{2}), (-1-\sqrt{2}, -1-\sqrt{2}, -1-\sqrt{2})$.

Definice 49. Je dáno $\mathfrak{i} = \langle f_1, ..., f_s\rangle \subset k[x_1, ..., x_n]$, l-tý *eliminační ideál* $\mathfrak{i}_l$ je ideál nad $k[x_{l+1}, ..., x_n]$ definovaný $\mathfrak{i}_l = \mathfrak{i} \cap k[x_{l+1}, ..., x_n]$.

Intuitivně se zdá, že $\mathfrak{i}_l$ obsahuje všechny prvky Gröbnerovy báze, ve kterých se vyskytují pouze proměnné $x_{l+1}, ..., x_n$. Eliminace proměnných znamená najít nenulové polynomy definující eliminační ideál $\mathfrak{i}_l$.

Věta 50 (Eliminační teorém). *Nechť $\mathfrak{i} \subset k[x_1, ..., x_n]$ je ideál a dále nechť G je Gröbnerova báze $\mathfrak{i}$ respektující lexikografické uspořádání $x_1 > x_2 > \cdots > x_n$. Pak pro každé $0 \leq \leq l \leq n$ množina $G_l = G \cap k[x_{l+1}, ..., x_n]$ je Gröbnerova báze l-tého eliminačního ideálu $\mathfrak{i}_l$.*

Pro ukázání, jak eliminační teorém funguje, použijeme následující příklad.

Příklad: Vezměme si příklad ze začátku této kapitoly.
$\mathfrak{i} = \langle x^2 + y + z - 1, x + y^2 + z - 1, x + y + z^2 - 1\rangle$. Dle eliminačního teorému dostáváme
$\mathfrak{i}_1 = \mathfrak{i} \cap \mathbb{C}[y, z] = \langle y^2 - y - z^2 + z, 2yz^2 + z^4 - z^2, z^6 - 4z^4 + 4z^3 - z^2\rangle$
$\mathfrak{i}_2 = \mathfrak{i} \cap \mathbb{C}[z] = \langle z^6 - 4z^4 + 4z^3 - z^2\rangle$.
Je zřejmé, že Gröbnerova báze při užití lexikografického uspořádání vyeliminuje nejen první proměnnou, ale dokonce první dvě, první tři proměnné, atd. Nevýhodou ale je značná časová náročnost výpočtu Gröbnerovy báze při lexikografickém uspořádání.

Nyní prodiskutujeme krok rozšíření. Předpokládejme, že máme ideál $\mathfrak{i} \subset k[x_1, ..., x_n]$. Máme afinní varietu $V(\mathfrak{i}) = \{(a_1, ..., a_n) \in k^n; f(a_1, ..., a_n) = 0; \forall f \in \mathfrak{i}\}$. K popisu bodů afinní variety musíme nejdříve vyřešit rovnici v jedné proměnné, kterou získáme eliminací zbývajících proměnných. Poté řešení postupně rozšiřujeme přidáním dalších proměnných.

Mějme nějaké l mezi 1 a n. Dostáváme eliminační ideál $\mathfrak{i}_l$ a řešení $(a_{l+1}, ..., a_n) \in V(\mathfrak{i}_l)$ označíme jako parciální řešení původního systému rovnic. K rozšíření na úplné řešení ve

$V(\mathfrak{i})$ nejdříve potřebujeme přidat k řešení jednu další proměnnou. To znamená najít a_l tak, že $(a_l, a_{l+1}, ..., a_n)$ leží ve varietě $V(\mathfrak{i}_{l-1})$ dalšího eliminačního ideálu. Konkrétněji předpokládejme, že $\mathfrak{i}_{l-1} = \langle g_1, ..., g_r \rangle$ v $k[x_l, x_{l+1}, ..., x_n]$. Chceme najít řešení $x_l = a_l$ rovnic $g_1(x_l, x_{l+1}, ..., x_n) = \cdots = g_r(x_l, x_{l+1}, ..., x_n) = 0$. Pracujeme s polynomy jedné proměnné x_l, tzn. že řešení a_l jsou právě kořeny největšího společného dělitele $g_1, ..., g_r$. Základním problémem je, když polynomy žádný společný kořen nemají, tzn existují parciální řešení, která nemůžeme rozšířit na úplné řešení. Např. uvažujme rovnice $xy = 1, xz = 1, \mathfrak{i} = \langle xy - 1, xz - 1 \rangle$ a jednoduchá aplikace eliminačního teorému nám dává, že $y - z$ generuje první eliminační ideál $\mathfrak{i}_1$. Tedy parciální řešení je tvaru (a, a) a rozšíření $(1, a, a)$ s výjimkou bodu $(0, 0)$.

Následující věta se zabývá problémem, kdy lze dané parciální řešení rozšířit na úplné řešení.

Věta 51 (**Teorém rozšíření**). *Nechť $\mathfrak{i} = \langle f_1, ..., f_s \rangle \subset \mathbb{C}[x_1, ..., x_n]$ a nechť $\mathfrak{i}_1$ je první eliminační ideál $\mathfrak{i}$. Pro každé $i; 1 \leq i \leq s$ píšeme f_i ve tvaru $f_i = g_i(x_2, ..., x_n)x_1^{N_i} +$ výrazy, kde x_1 má stupeň $< N_i$, kde $N_i \geq 0, g_i \in \mathbb{C}[x_2, ..., x_n]$ je nenulový. Předpokládejme, že máme parciální řešení $(a_2, ..., a_n) \in V(\mathfrak{i}_1)$. Jestliže $(a_2, ..., a_n) \notin V(g_1, ..., g_s)$, pak existuje $a_1 \in \mathbb{C}$ tak, že $(a_1, ..., a_n) \in V(\mathfrak{i})$.*

Věta je definována pro $k = \mathbb{C}$. Ukažme si následující příklad.

Příklad: Nechť $k = \mathbb{R}$. Mějme soustavu rovnic $x^2 = y, x^2 = z$.
Eliminujeme x a dostaneme rovnici $y = z$ a tedy parciální řešení jsou $(a, a); \forall a \in \mathbb{R}$. Pokud $k = \mathbb{C}$, pak můžeme z rovnic ihned dopočítat úplná řešení soustavy. Rozšířit řešení na úplná řešení nad $\mathbb{R}$ někdy nelze. V našem případě $x = a^2$ nastává problém pro $a < 0$, kdy rovnice nemá řešení. Odtud vidíme, že předchozí věta neplatí pro $k = \mathbb{R}$.

Tvrzení 52. *Nechť $\mathfrak{i} = \langle f_1, ..., f_s \rangle \subset \mathbb{C}[x_1, ..., x_n]$ a předpokládejme, že pro nějaké i je f_i tvaru $f_i = cx_1^N +$ výraz, kde x_1 má stupeň $< N$, kde $c \in \mathbb{C}$ je nenulové a $N > 0$. Jestliže $\mathfrak{i}_1$ je první eliminační ideál $\mathfrak{i}$ a $(a_2, ..., a_n) \in V(\mathfrak{i}_1)$, pak existuje $a_1 \in \mathbb{C}$ tak, že $(a_1, ..., a_n) \in V(\mathfrak{i})$.*

Příklad: Mějme soustavu rovnic

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

Redukovaná Gröbnerova báze ideálu $\mathfrak{i} = \langle x^2 + 2y^2 - 3, x^2 + xy + y^2 - 3 \rangle$ vzhledem k lexikografickému uspořádání pro $x > y$ je $G = \{y^3 - y, xy - y^2, x^2 + 2y^2 - 3\}$. První eliminační ideál $\mathfrak{i}_1 = \mathfrak{i} \cap k[x] = \langle g_1 \rangle = \langle y^3 - y \rangle$. Kořeny g_1 jsou $y_1 = 0, y_2 = 1, y_3 = -1$. Postupně dosadíme do zadání a získáme tak úplná řešení $\left[-\sqrt{3}, 0\right], \left[\sqrt{3}, 0\right], [1, 1], [-1, -1]$. Tímto jsme získali čtyři přesná řešení soustavy rovnic.

2.3.2. Implicitizace

Implicitizace znamená převod parametrického vyjádření afinních variet na implicitní vyjádření. Implicitizaci můžeme řešit pomocí Gröbnerových bází při použití lexikografického uspořádání.

Nejdříve uveďme parametrizaci zadanou pomocí polynomů, tzv. polynomickou parametrizaci. Můžeme ji vyjádřit ve tvaru

$$\begin{aligned} x_1 &= f_1(t_1, ..., t_m), \\ &\vdots \\ x_n &= f_n(t_1, ..., t_m), \end{aligned}$$

kde $f_1, ..., f_n$ jsou polynomy v $k[t_1, ..., t_m]$. Geometricky představuje soustava zobrazení $F : k^m \to k^n$ definované $F(t_1, ..., t_m) = (f_1(t_1, ..., t_m), ..., f_n(t_1, ..., t_m))$. Poté $F(k^m) \subset k^n$. Jelikož $F(k^m)$ nemusí být afinní varietou, řešením převodu parciálního vyjádření na implicitní je nalézt nejmenší variety obsahující $F(k^m)$.

Úkol implicitizace tedy spočívá ve vyloučení parametrů z parametrického vyjádření. Výsledné rovnice pak obsahují pouze proměnné $x_1, ..., x_n$. Eliminaci můžeme provést pomocí výpočtu redukované Gröbnerovy báze ideálu $\mathfrak{i} = \langle x_1 - f_1, .., x_n - f_n \rangle$.

Věta 53 (Polynomická implicitizace). *Nechť k je nekonečné pole a $F : k^m \to k^n$ je funkce definovaná polynomickou parametrizací. Nechť $\mathfrak{i}$ je ideál $\mathfrak{i} = \langle x_1 - f_1, ..., x_n - f_n \rangle \subset k[t_1, ..., t_m, x_1, ..., x_n]$ a nechť $\mathfrak{i}_m = \mathfrak{i} \cap k[x_1, ..., x_n]$ je m-tý eliminační ideál. Pak $V(\mathfrak{i}_m)$ je nejmenší varieta v k^n obsahující $F(k^m)$.*

Příklad: Uvažujme křivku zadanou parametricky $x = t, y = t^2, z = t^3$. Plochu tečen křivky pak můžeme vyjádřit ve tvaru $x = t + u, y = t^2 + 2tu, z = t^3 + 3t^2u$. Použijeme algoritmus na převod na implicitní vyjádření. Dostaneme tak redukovanou Gröbnerovu bázi, z které vybereme takový prvek, který neobsahuje ani t ani u. V našem případě je tvaru $x^3z - \frac{3}{4}x^2y^2 - \frac{3}{2}xyz + y^3 + \frac{1}{4}z^2 = 0$, což je implicitní vyjádření dané plochy.

Druhým případem je racionální implicitizace. Zde může nastat pár problémů.

Příklad: Mějme racionální parametrizaci plochy $x = \frac{u^2}{v}, y = \frac{v^2}{u}, z = u$. Snadno ověříme, že libovolný bod (x, y, z) splňující zadání, leží na ploše $x^2y = z^3$. Odstraníme zlomky a převedeme na implicitní vyjádření pro ideál $\mathfrak{i} = \langle vx - u^2, uy - v^2, z - u \rangle \subset k[u, v, x, y, z]$. Vyjádříme druhý eliminační ideál $\mathfrak{i}_2 = \mathfrak{i} \cap k[x, y, z] = \langle z(x^2y - z^3) \rangle$ a tedy $V(\mathfrak{i}) = V(x^2y - z^3) \cup V(z)$.
Odtud vidíme, že do výsledku se přidala celá rovina $z = 0$ a tedy $V(\mathfrak{i}_2)$ není nejmenší varietou obsahující danou parametrizaci. Tento problém nastává kvůli odstranění zlomků, které musíme provést lépe, zajištěním nenulovosti jmenovatelů. Ideál $\mathfrak{i}$ můžeme upravit tak, že do něj přidáme jednu proměnnou a jednu rovnici, která zajistí nenulovost jmenovatelů. Ideál $\mathfrak{i}$ tedy nahradíme ideálem $\mathfrak{j} = \langle vx - u^2, uy - v^2, z - u, 1 - w(uv) \rangle \subset k[w, u, v, x, y, z]$, kde rovnice $1 - w(uv) = 0$ zajišťuje nenulovost u, v ve všech bodech $V(\mathfrak{j})$. Třetí eliminační ideál je tvaru $\mathfrak{j}_3 = \mathfrak{j} \cap k[x, y, z] = \langle x^2y - z^3 \rangle$.

Racionální parametrizaci můžeme obecně vyjádřit ve tvaru

$$x_1 = \frac{f_1(t_1, ..., t_m)}{g_1(t_1, ..., t_m)},$$
$$\vdots$$
$$x_n = \frac{f_n(t_1, ..., t_m)}{g_n(t_1, ..., t_m)},$$

kde $f_1, g_1, ..., f_n, g_n$ jsou polynomy v $k[t_1, ..., t_m]$. Zobrazení $F : k^m \to k^n$ ale nemůžeme definovat na celém k^m, protože musíme vyjmout ty body $(t_1, ..., t_m)$, pro které $g_i(t_1, ..., t_m) = 0$ pro nějaké i. Označíme $W = V(g_1, ..., g_n) \subset k^m$, pak $F(t_1, ..., t_m) = (\frac{f_1(t_1,...,t_m)}{g_1(t_1,...,t_m)}, \cdots, \frac{f_n(t_1,...,t_m)}{g_n(t_1,...,t_m)})$ definuje zobrazení $F : k^m - W \to k^n$. Cíl je nalézt nejmenší varietu v k^n obsahující $F(k^m - W)$.

Věta 54 (**Racionální implicitizace**). *Nechť k je nekonečné pole a $F : k^m - W \to k^n$ je funkce definovaná racionální parametrizací. Nechť $\mathfrak{j}$ je ideál $\mathfrak{j} = \langle g_1x_1 - f_1, ..., g_nx_n - f_n, 1 - gy\rangle \subset k[y, t_1, ..., t_m, x_1, ..., x_n]$, kde $g = g_1g_2...g_n$, nechť $\mathfrak{j}_{m+1} = \mathfrak{j} \cap k[x_1, ..., x_n]$ je $(m+1)$-ní eliminační ideál. Pak $V(\mathfrak{j}_{m+1})$ je nejmenší varieta v k^n obsahující $F(k^m - W)$.*

Tato věta ukazuje, jak se racionální parametrizace převede na implicitní vyjádření. Odstraníme zlomky vynásobením i-té rovnice funkcí g_i a tím, že přidáme rovnici $1 - g_1...g_ny = 0$ zajistíme nenulovost $g_1, ..., g_n$ na dané varietě. Poté vypočteme redukovanou Gröbnerovu bázi ideálu $\mathfrak{j}$ vzhledem k lexikografickému uspořádání pro $y > t_1 > \cdots > t_m > x_1 > \cdots > x_n$. Členy Gröbnerovy báze neobsahující žádnou z proměnných t_j, t_i definují implicitní vyjádření afinní variety.

Příklad: Mějme parametrické vyjádření Descartova listu $x = \frac{3at}{1+t^3}, y = \frac{3at^2}{1+t^3}$. Předcházejícím algoritmem dostaneme ideál $\mathfrak{i} = \langle x(1 + t^3) - 3at, y(1 + t^3) - 3at^2.1 - w(1 + t^3)\rangle \subset k[w, t, x, y]$. Prvek redukované Gröbnerovy báze neobsahující proměnnou w ani proměnnou t je tvaru $x^3 - 3axy + y^3 = 0$, což je implicitní vyjádření Descartova listu.

Nyní se podívejme na další možnosti parametrizace. Po jistých úpravách totiž můžeme použít Gröbnerovy báze i pro nalezení implicitního vyjádření některých afinních variet, které jsou parametrizovány pomocí goniometrických funkcí. Musíme zavést označení příslušných funkcí, např. $c_t = \cos t, s_t = \sin t$, z čehož vidíme, že dostaneme polynomy v proměnných c_t, s_t. Dále musíme přidat identitu $c_t^2 + s_t^2 = 1$, jinak bychom měli málo rovnic a nemohli bychom eliminovat všechny parametry. dále už postupujeme stejně jako v předchozích příkladech.

Příklad: Parametrické vyjádření Bernoulliovy lemniskáty je tvaru $x = \frac{a\cos t}{1+\sin^2 t}$, $y = \frac{a\cos t\sin t}{1+\sin^2 t}$. Nechť $c_t = \cos t, s_t = \sin t$. Poté dostáváme rovnice tvaru $x(1 + s_t^2) - ac_t = 0, y(1 + s_t^2) - ac_ts_t = 0$. K těmto rovnicím přidáme identitu $c_t^2 + s_t^2 = 1$. Nyní použijeme algebraického převodu polynomické parametrizace na implicitní vyjádření, jelikož jmenovatel není nikdy roven nule. Spočítáme redukovanou Gröbnerovu bázi a vybereme opět ten prvek, který neobsahuje ani proměnnou c_t ani proměnnou s_t. V našem případě je tvaru $x^4 + 2x^2y^2 - a^2x^2 + y^4 + a^2y^2 = 0$. Tuto rovnici můžeme přepsat do tvaru $(x^2 + y^2)^2 - a^2(x^2 - y^2)^2 = 0$, což je implicitní vyjádření Bernoulliovy lemniskáty.

3. Kryptografie

3.1. Multivariační kryptosystémy

Tato kapitola je inspirována [6].

Podstatnou změnou moderního komunikačního systému byla revoluční myšlenka kryptosystému s veřejným klíčem. Ta byla poprvé uvedena Diffiem a Hellmanem. První praktická realizace veřejného kryptosystému je známý RSA kryptosystém od Rivesta, Shamira a Adlemana.

V kryptosystémech s veřejným klíčem se klíč skládá ze 2 různých částí, veřejné a tajné. Veřejný klíč je dostupný všem a je používán k zašifrování zprávy nebo k ověření autentičnosti elektronického podpisu. Tajný klíč je používán k rozšifrování zašifrované zprávy nebo k tvorbě elektronického podpisu. Tato asymetrie nám umožňuje bezpečně komunikovat skrz veřejný komunikační kanál bez předchozí změny tajného klíče. Pro symetrické kryptosystémy musí dva lidé, kteří spolu chtějí bezpečně komunikovat, mít stejný (symetrický) klíč a oba se musí na tomto klíči dohodnout dříve, nebo používají protokol na změnu veřejných klíčů.

3.1.1. Typy multivariačních kryptosystémů

Existující multivariační kryptosystémy můžeme rozdělit na explicitní kryptosystémy a implicitní kryptosystémy. Oba typy můžeme použít pro zakódování i pro elektronický podpis. Pro zašifrování musí být všechna zobrazení invertibilní, abychom z dané zašifrované zprávy mohli jednoznačně najít zprávu původní. U podpisu hledáme, zda se shoduje s některým z několika možných vzorů.

Můžeme použít $X = (x_1, ..., x_n)$ k označení klasického souřadnicového systému v k^n, $Y = (y_1, ..., y_m)$ v k^m, kde k je vhodné konečné pole. Pro zašifrování použijeme $\acute{X} = (\acute{x}_1, ..., \acute{x}_n)$ k označení prvku v k^n, který budeme označovat jako nešifrovaný text (nešifrovaná tajná zpráva) a $\acute{Y} = (\acute{y}_1, ..., \acute{y}_m)$ označíme prvek v k^m a nazveme ho zašifrovaný text (zašifrovaná tajná zpráva). V případě elektronického podpisu použijeme $\acute{Y} = (\acute{y}_1, ..., \acute{y}_m)$ k označení prvku v k^m jako zprávy a $\acute{X} = (\acute{x}_1, ..., \acute{x}_n)$ k označení prvku v k^n, což je elektronický podpis zprávy $\acute{Y}$.

Explicitní systémy

V explicitním multivariačním kryptosystému s veřejným klíčem máme zobrazení $F : k^n \to k^m$ takové, že

$$F(x_1, .., x_n) = (F_1(x_1, ..., x_n), ..., F_m(x_1, ..., x_n)) = (y_1, ..., y_m) = Y,$$

kde $F_i(x_1, ..., x_n)$ je polynom v $x_1, ..., x_n$.

Konstrukce klíče pro tento typ systému je následující. Nejprve vytvoříme zobrazení $f : k^n \to k^m$ takové, že

$$f(x_1, ..., x_n) = (f_1(x_1, ..., x_n), ..., f_m(x_1, ..., x_n),$$

kde $f_i(x_1, ..., x_n)$ je polynom v $x_1, ..., x_n$ a rovnice

$$f(x_1, ..., x_n) = (f_1(x_1, ..., x_n), ..., f_m(x_1, ..., x_n)) = (a_1, ..., a_m)$$

je jednoduše řešitelná. Jinými slovy snadno najdeme předobraz f. Označme f^{-1} hledání předobrazu. Pak F zkonstruujeme jako

$$F = L_1 \circ f \circ L_2, \tag{3.1}$$

kde L_1 je náhodně vybrané afinní invertibilní lineární zobrazení z k^m do k^m, $L_1(x_1, ..., x_m) = X \times A_1 + C_1, A_1$ je $m \times m$ invertibilní matice a $C_1 \in k^m$, L_2 je (afinní) invertibilní lineární zobrazení z k^n do k^n, $L_2(x_1, ..., x_n) = X \times A_2 + C_2, A_2$ je $n \times n$ invertibilní matice a $C_2 \in k^n$.

V tomto případě se veřejný klíč skládá z m polynomů F a struktury pole k. Tajný klíč se skládá převážně z L_1 a L_2. Idea klíče je, že L_1, L_2 slouží ke skrytí zobrazení f, které je snadno řešitelné. V některých systémech může být funkce f známá, kdežto v ostatních systémech je f tajná. Za účelem zašifrování zprávy $\acute{X}$ spočítáme $F(\acute{X})$. K rozšifrování zprávy $\acute{Y}$ musíme vyřešit rovnici

$$F(x_1, ..., x_n) = \acute{Y}. \tag{3.2}$$

V případě elektronického podpisu, k podepsání zprávy $\acute{Y}$ musíme vyřešit rovnici (3.2), jejíž řešení označíme $\acute{X}$. K ověření legitimity podpisu potřebujeme zkontrolovat, zda je splněno $F(\acute{x}_1, ..., \acute{x}_n) = \acute{Y}$. Dle tohoto postupu vidíme, že můžeme najít předobraz $\acute{Y}$ aplikací $(L_1)^{-1}, f^{-1}, (L_2)^{-1}$.

Implicitní systémy

V implicitním multivariačním kryptosystému s veřejným klíčem máme množinu l polynomů tvaru

$$\begin{aligned} H(X,Y) &= H(x_1, ..., x_n, y_1, ..., y_m) = \\ (H_1(x_1, ..., x_n, y_1, ..., y_m), &..., H_l(x_1, ..., x_n, y_1, ..., y_m)) = (0, ..., 0), \end{aligned} \tag{3.3}$$

kde $H_i(x_1, ..., x_n, y_1, ..., y_m)$ je polynom v $x_1, ..., x_n, y_1, ..., y_m$.

Při konstrukci klíče nejdříve zkonstruujeme rovnice tvaru

$$h(X,Y) = h(x_1, ..., x_n, y_1, ..., y_m) = (h_1(x_1, ..., y_m), ..., h_l(x_1, ..., y_m)) = (0, ..., 0),$$

kde $h_i(x_1, ..., y_m)$ je polynom v $x_1, ..., y_m$. Musíme splnit následující dva požadavky

- Pro daný prvek $\acute{X}$ lze snadno vyřešit rovnice

$$h(\acute{x}_1, ..., \acute{x}_n, y_1, ..., y_m) = (0, ..., 0), \tag{3.4}$$

  jejíž řešení označíme $\acute{Y} = (\acute{y}_1, ..., \acute{y}_m)$ a

- pro daný prvek $\acute{Y}$ lze snadno vyřešit rovnice

$$h(x_1, ..., x_n, \acute{y}_1, ..., \acute{y}_m) = (0, ..., 0), \tag{3.5}$$

  jejíž řešení označíme $\acute{X} = (\acute{x}_1, ..., \acute{x}_n)$.

Ve většině případech je (3.4) ve skutečnosti množina lineárních rovnic a (3.5) je množina speciálně zkonstruovaných nelineárních rovnic.

Rovnici H poté zkonstruujeme jako

$$H = L_3 \circ h(L_2(X), L_1(Y)) = (0, ..., 0),$$

ke L_1, L_2 jsou definovány stejně jako v explicitním případě a L_3 je invertibilní lineární zobrazení z k^l do k^l.

Pro šifrování zprávy $\acute{X}$, $\acute{X}$ vstupuje do rovnic (3.4). Pak řešíme rovnici

$$H(\acute{X}, Y) = H(x_1, ..., x_n, y_1, ..., y_n) = (0, ..., 0),$$

a řešení označíme $\acute{Y}$, což je zašifrovaná zpráva, tedy šifrovaný text.

Při rozšifrování zprávy $\acute{Y}$ musíme nejdříve spočítat $\acute{\bar{Y}} = L_2^{-1}(\acute{Y})$, pak přidáme $\acute{\bar{Y}}$ do rovnic (3.5). Následně řešíme rovnici

$$h(X, \acute{\bar{Y}}) = h(x_1, ..., x_n, \acute{\bar{y}}_1, ..., \acute{\bar{y}}_m) = (0, ..., 0),$$

řešení označíme $\acute{\bar{y}}$. Nešifrovaný text je dán $\acute{Y} = (L_2)^{-1}(\acute{\bar{y}})$.

Pro elektronický podpis, tedy k podepsání zprávy $\acute{Y}$ musíme projít dešifrovacím procesem, abychom našli prvek $\acute{x}$ v k^n. K ověření pravosti podpisu potřebujeme ověřit, že

$$H(\acute{x}_1, ..., \acute{x}_n, \acute{y}_1, ..., \acute{y}_m) = (0, ..., 0).$$

V tomto případě se veřejný klíč skládá z polynomů v H a struktury pole k. Tajný klíč se skládá především z L_1, L_2, L_3. Myšlenkou klíče je opět ukrytí rovnice $h(X, Y) = 0$, která může být snadno řešitelná pro danou hodnotu Y. Ukrytí realizujeme pomocí L_1, L_2, L_3.

3.1.2. Základní bezpečnost

Nejdůležitějším faktorem pro multivariační kryptosystémy je jejich bezpečnost a efektivita. Prodiskutujeme základní aspekty těchto požadavků v kontextu zašifrovacích systémů.

V každém šifrovacím procesu používáme zobrazení z k^n do k^m na prvek v k^n. V rozšifrovacím procesu hledáme jeho "inverzi", tedy řešíme rovnici (3.2). To znamená, že rovnice (3.2) musí být obtížně řešitelná. Má-li zašifrování inverzi, kterou můžeme vyjádřit jako polynomické zobrazení, pak musíme zabezpečit, aby toto inverzní zobrazení bylo velmi vysokého stupně, jinak by kdokoli mohl použít veřejný klíč ke generování dostatečného počtu párů šifrovaných a nešifrovaných textů a najít tak lehce inverzi. Ze samotné konstrukce musíme zajistit, abychom pouze obtížně faktorizovali zašifrovací zobrazení ve tvaru (3.1). To je obecně obtížné, protože faktorizace multivariačních zobrazení je extrémně těžká.

Jistě je každý kryptosystém s veřejným klíčem určen pro praktické aplikace. To vyžaduje, aby byl proces šifrování a dešifrování účinný. Veřejným klíčem je množina multivariačních polynomů, která musí být nejprve přenesena a uložena a pak musí být spočteny hodnoty těchto polynomů. Tedy tyto polynomy F_i musí být malého stupně (ale ne lineární, jinak by byl systém nepoužitelný). Tedy nejlepší volbou jsou kvadratické polynomy.

3.1.3. Explicitní systémy

Triangulární kryptosystémy

Triangulární kryptosystémy vynalezli Diffie a Fell. Jejich myšlenka byla vytvořit kryptosystém užitím skládání mnoha invertibilních lineárních zobrazení a triangulárních zobrazení tvaru

$$T(x_1, ..., x_n) = (x_1 + g(x_2, ..., x_n), x_2, ..., x_n), \tag{3.6}$$

kde g_i je polynom. Zřejmě T je invertibilní a navíc může být aplikován proces rozšifrování. Vidíme, že není žádný způsob, jak zkonstruovat systém takový, že je bezpečný a má veřejný klíč praktické velikosti.

Triangulární zobrazení patří k třídě de Jonquierèových zobrazení

$$J(x_1, ..., x_n) = (x_1 + g_1(x_2, ..., x_n), x_2 + g_2(x_3, ..., x_n), ..., x_{n-1} + g_{n-1}(x_n), x_n), \tag{3.7}$$

kde g_i jsou polynomické funkce.

Zřejmě J je snadno invertovatelná. Všechna invertibilní afinní lineární zobrazení nad k^n a de Jonquierèova zobrazení jsou v algebraické geometrii nazvána krotkou transformací. Krotké transformace jsou prvky grupy automorfismu polynomického okruhu $k[x_1, ..., x_n]$. Prvky, které jsou v této grupě a nejsou krotké, jsou označeny jako divoké.

Vidíme, že de Jonquierèova zobrazení jsou dvojího typu, horní triangulace a dolní triangulace. Konstrukce kvadratického zobrazení f je dána

$$f = J_u \circ J_l \circ I(x_1, ..., x_n). \tag{3.8}$$

Zde J_u je horní triangulární de Jonquierèovo zobrazení v k^m a J_l dolní triangulární de Jonquierèovo zobrazení v k^m a lineární zobrazení I je vnoření k^n do $k^m : I(x_1, ..., x_n) = (x_1, ..., x_n, 0, ..., 0)$. Největší přínos této konstrukce je, že f je kvadratická funkce a že každá lineární kombinace složek f neprodukuje lineární funkci. Trik této konstrukce je užití zobrazení I. Vidíme, že

$$J_l \circ I(x_1, ..., x_n) = (x_1, x_2 + g_1(x_1), ..., x_n + g_{n-1}(x_1, ..., x_{n-1}), g_n(x_1, ..., x_n), ..., g_{m-1}(x_1, ..., x_n)),$$

což nám dává volnost ve výběru všech $g_i, i = n, ..., m-1$. Tato metoda je nazvána krotkou transformační metodou (TTM). Navzdory tomu, že autor tvrdil, že TTM systémy jsou velmi bezpečné pro všechny obvyklé útoky, krátce na to Curtois a Goubin užili Minrank metodu na útok na tento systém. Tato metoda vyhledává matice nejnižší hodnosti z prostoru lineárního z několika daných matic. Na TTM kryptosystémy můžeme také použít Patarinovu linearizační metodu.

Matsumoto-Imai systémy

Další myšlenku, jak zkonstruovat multivariační kryptosystémy, předložili Matsumoto a Imai. Zde ideou klíče je užít zobrazení $\bar{f}$ nad rozšířeným polem $\bar{K}$ stupně rozšíření n konečného pole k (s charakteristikou 2). Zobrazení ϕ identifikuje $\bar{K}$ jako k^n, poté identifikuje toto zobrazení jako multivariační polynomické zobrazení $f : k^n \to k^n$:

$$f = \phi \circ \bar{f} \circ \phi^{-1}. \tag{3.9}$$

Pak můžeme "skrýt" toto zobrazení f skládáním obou stran rovnice dvěma invertibilními afinními lineárními zobrazeními L_1, L_2 v k^n. Zobrazení $\bar{f}$ je navrženo Matsumotem a Imaiem jako zobrazení

$$f : X \mapsto X^{1+q^i}, \tag{3.10}$$

kde q je počet prvků v k, X je prvek $\bar{K}$, k je charakteristiky 2 tak, že $GCD(1-q^i, q^n-1) = 1$. Poslední podmínka zajistí, že zobrazení $\bar{f}$ lze jednoduše invertovat. Inverze zobrazení $\bar{f}$ je dána

$$\bar{f}^{-1}(X) = X^t, \tag{3.11}$$

kde $t(1+q^i) = 1 \; mod \; (q^n - 1)$. To zajistí, že můžeme rozšifrovat každou tajnou zprávu jednoduše pomocí inverze. Důležité je, že f je kvadratické, dle vlastnosti Frobeniova zobrazení $X \mapsto X^{q^i}$.

Metoda skrytého pole rovnic (HFE)

Tato metoda je navržena Patarinem jako nejsilnější metoda. V tomto případě je rozdíl oproti původnímu Matsumoto-Imai systému ten, že $\bar{f}$ je nahrazeno obecnějším zobrazením

$$\bar{f} : X \mapsto \sum_{i,j}^{A} a_{ij} X^{q^i+q^j} + \sum_{i}^{A} b_i X^{q^i} + C, \tag{3.12}$$

kde koeficienty jsou vybrány náhodně. Proces rozšifrování zahrnuje řešení rovnic $\bar{f} = \acute{Y}$ pro X.

3.1.4. Implicitní systémy

Implicitní systémy nejsou tak rozšířené jako explicitní. Existují dvě třídy implicitních systémů nazvané Malý drak a Drak. Malý drak je zjednodušená verze draka. Tyto konstrukce jsou inspirovány linearizací rovnic a kryptosystémy Matsumoto-Imai jsou v podstatě kombinací těchto dvou myšlenek.

3.2. Algoritmy

3.2.1. Buchbergerův algoritmus

Buchbergerův algoritmus

Příklad: Uvažujme pole $k[x, y]$ se stupňovaným lexikografickým uspořádáním a nechť $\mathfrak{i} = \langle f_1, f_2 \rangle = \langle x^3 - 2xy, x^2y - 2y^2 + x \rangle$. Poznamenejme, že $\{f_1, f_2\}$ není Gröbnerova báze $\mathfrak{i}$, protože $LT(S(f_1, f_2)) = -x^2 \notin \langle LT(f_1), LT(f_2) \rangle$.

Pro vytvoření Gröbnerovy báze vyvstává přirozená myšlenka zkusit nejdříve rozšířit původní generující množinu přidáním více polynomů do $\mathfrak{i}$. V určitém smyslu nám to nedá nic nového, pouze to přináší redundanci do báze $\mathfrak{i}$. Otázkou je, které další generátory musíme do generující množiny přidat.

Pro S-polynom $S(f_1, f_2) = -x^2 \notin \mathfrak{i}$ je zbytek po dělení $F = \{f_1, f_2\}$ roven $-x^2$, což není rovno nule a tudíž ho přidáme do generující množiny a označíme ho f_3. Poté ověříme, zda $F = \{f_1, f_2, f_3\}$ je již Gröbnerova báze. Tedy spočítáme všechny S-polynomy.
$S(f_1, f_2) = f_3 \Rightarrow \overline{S(f_1, f_2)}^F = 0$
$S(f_1, f_3) = (x^3 - 2xy) - (-x)(-x^2) = -2xy \Rightarrow \overline{S(f_1, f_3)}^F = -2xy \neq 0.$
Do generující množiny tedy přidáme $f_4 = -2xy$. Pokračujeme dále v ověření
$\overline{S(f_1, f_2)}^F = \overline{S(f_1, f_3)}^F = 0$
$S(f_1, f_4) = -2xy^2 = yf_4 \Rightarrow \overline{S(f_1, f_4)}^F = 0$
$S(f_2, f_3) = -2y^2 + x \Rightarrow \overline{S(f_2, f_3)}^F = -2y^2 + x \neq 0.$
Musíme tedy opět rozšířit generující množinu o $f_5 = -2y^2 + x$ a nyní se již snadno ověří, že $\{f_1, ..., f_5\} = \{x^3 - 2xy, x^2y - 2y^2 + x, -x^2, -2xy, -2y^2 + x\}$ je Gröbnerova báze pro ideál $\mathfrak{i}$.

Věta 55 (**Buchbergerův algoritmus**). *Nechť* $\mathfrak{i} = \langle f_1, ..., f_s \rangle \neq \{0\}$ *je polynomický ideál. Pak Gröbnerova báze pro* $\mathfrak{i}$ *může být zkonstruována v konečném počtu kroků následujícího algoritmu*
INPUT: $F = \langle f_1, ..., f_s \rangle$
OUTPUT: Gröbnerova báze $G = (g_1, ..., g_t)$ *ideálu* $\mathfrak{i}$*, kde* $F \subset G$
$G := F$
REPEAT
$\widetilde{G} := G$
FOR každou dvojici $\{p, q\}; p \neq q$ *v* $\widetilde{G}$ *DO*
$S := \overline{S(p,q)}^{\widetilde{G}}$
IF $S \neq 0$ *THEN* $G := G \cup \{S\}$
UNTIL $G = \widetilde{G}$

Gröbnerova báze spočítaná užitím Buchbergerova algoritmu je často větší, než je nezbytné. Přebytečné generátory můžeme eliminovat užitím následujícího faktu.

Lemma 56. *Nechť* G *je Gröbnerova báze polynomického ideálu* $\mathfrak{i}$*. Nechť* $p \in G$ *je polynom splňující* $\langle LT(p) \rangle \in \langle LT(G - \{p\}) \rangle$*. Pak* $G - \{p\}$ *je také Gröbnerova báze ideálu* $\mathfrak{i}$*.*

Důkaz. Víme, že $\langle LT(G) \rangle = \langle LT(\mathfrak{i}) \rangle$. Jestliže $LT(p) \in \langle LT(G - \{p\}) \rangle$, pak dostáváme $\langle LT(G - \{p\}) \rangle = \langle LT(G) \rangle$. Podle definice plyne, že $G - p$ je také Gröbnerova báze ideálu $\mathfrak{i}$. □

Definice 57. *Minimální Gröbnerova báze* polynomického ideálu $\mathfrak{i}$ je Gröbnerova báze G splňující
i) $LC(p) = 1; \forall p \in G$
ii) $\forall p \in G; LT(p) \notin \langle LT(G - \{p\}) \rangle$.

Minimální Gröbnerovu bázi daného nenulového ideálu tedy sestrojíme užitím Buchbergerova algoritmu a následným použitím předešlého lemmatu.

Příklad: Vzhledem ke stupňovanému lexikografickému uspořádání jsme již spočetli, že Gröbnerova báze $\{f_1, ..., f_5\} = \{x^3 - 2xy, x^2y - 2y^2 + x, -x^2, -2xy, -2y^2 + x\}$. Vidíme, že některé hlavní koeficienty jsou různé od 1, tudíž musíme generátory vynásobit vhodnými konstantami. Do minimální Gröbnerovy báze pak nezařadíme polynom f_1, jelikož $LT(f_1) = x^3 = -xLT(f_3)$. Podobně ani f_2 nebude členem Gröbnerovy báze, jelikož $LT(f_2) = x^2y = -\frac{1}{2}xLT(f_4)$. Dále již nenalezneme další podobný případ, kdy hlavní člen generátoru dělí hlavní člen jiného generátoru. Dostáváme tak minimální Gröbnerovu bázi tvořenou polynomy $\widehat{f_3} = x^2, \widehat{f_4} = xy, \widehat{f_5} = y^2 - \frac{1}{2}x$.

Ideál uvedený v předchozím případě může mít více minimálních Gröbnerových bází (např. $\widetilde{f_3} = x^2 + axy, \widetilde{f_4} = \widehat{f_4}, \widetilde{f_5} = \widehat{f_5}$). Tedy může existovat i nekonečný počet minimálních Gröbnerových bází. Umíme ale vybrat takovou bázi, která je v určitém smysle lepší, než báze zbývající.

Definice 58. *Redukovaná Gröbnerova báze* polynomického ideálu $\mathfrak{i}$ je Gröbnerova báze G taková, že
i) $LC(p) = 1; \forall p \in G$
ii) $\forall p \in G$, žádný monom p neleží v $\langle LT(G - \{p\}) \rangle$.

Tvrzení 59. *Nechť* $\mathfrak{i} \neq \{0\}$ *je polynomický ideál. Pak pro dané monomické uspořádání má* $\mathfrak{i}$ *jedinou redukovanou Gröbnerovu bázi.*

Uveďme nyní příklad, který demonstruje souvislost Buchbergerova algoritmu a Gaussovy eliminační metody.

Příklad: Mějme soustavu rovnic

$$\begin{aligned} 3x - 6y - 2z &= 0 \\ 2x - 4y + 4w &= 0 \\ x - 2y - z - w &= 0 \end{aligned}$$

Použijeme Gaussovu eliminaci na matici koeficientů soustavy a dostaneme tvar

$$\begin{pmatrix} 1 & -2 & -1 & -1 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Abychom získali redukovanou matici, musí být každá hlavní jednička jedinou nenulovou hodnotou ve sloupci. Tedy dostáváme matici

$$\begin{pmatrix} 1 & -2 & 0 & 2 \\ 0 & 0 & 1 & 3 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Nechť $\mathfrak{i} = \langle 3x - 6y - 2z, 2x - 4y + 4w, x - 2y - z - w \rangle \subset k[x, y, z, w]$ je ideál odpovídající zadané soustavě lineárních rovnic. Minimální gröbnerova báze vzhledem k lexikografickému uspořádání $x > y > z > w$ je tvaru $\mathfrak{i} = \langle x - 2y - z - w, z + 3w \rangle$, což odpovídá první matici. Redukovaná Gröbnerova báze ideálu $\mathfrak{i}$ je tvaru $\mathfrak{i} = \langle x - 2y + 2w, z + 3w \rangle$, což odpovídá druhé matici.

Vylepšení Buchbergerova algoritmu

Základní Buchbergerův algoritmus je početně dosti náročný a to především výpočet S-polynomu a následné dělení, při kterém zjišťujeme zbytek po dělení prvky báze. Proto se nyní budeme věnovat tomu, jak tento algoritmus vylepšit a zkrátit tak dobu výpočtu. Snahou je najít takové S-polynomy, které nemusíme při dělení uvažovat.

Definice 60. Mějme monomické uspořádání a nechť $G = \{g_1, ..., g_t\} \subset k[x_1, ..., x_n]$. Je dán $f \in k[x_1, ..., x_n]$, řekneme, že f se redukuje na nulu modulo G a značíme $f \underset{G}{\rightarrow} 0$, pokud f můžeme zapsat ve tvaru $f = a_1 g_1 + \cdots + a_t g_t$, je- li $a_i g_i \neq 0$, pak máme $multideg(f) \geq multideg(a_i g_i)$.

Vztah mezi redukcí na nulu modulo G a algoritmem dělení množinou polynomů G popisuje následující lemma.

Lemma 61. *Nechť* $G = (g_1, ..., g_t)$ *je uspořádaná množina prvků* $k[x_1, ..., x_n]$ *a je dán* $f \in k[x_1, ..., x_n]$. *Pak* $\overline{f}^G = 0 \Rightarrow f \underset{G}{\rightarrow} 0$. *Obrácené tvrzení obecně neplatí.*

Věta 62. *Báze $G = \{g_1, ..., g_t\}$ ideálu $\mathfrak{i}$ je Gröbnerova báze právě tehdy, když $S(g_i, g_j) \underset{G}{\rightarrow} 0$ pro $\forall i, j; i \neq j$.*

Tvrzení 63. *Je dána konečná množina $G \subset k[x_1, ..., x_n]$, předpokládejme, že máme $f, g \in G$ tak, že $LCM(LM(f), LM(g)) = LM(f)LM(g)$. Potom $S(f, g) \underset{G}{\rightarrow} 0$.*

Příklad: Mějme $G = \langle yz + y, x^3 + y, z^4 \rangle$ se stupňovaným lexikografickým uspořádáním na $k[x, y, z]$. Pak $S(x^3 + y, z^4) \underset{G}{\rightarrow} 0$ podle předchozího tvrzení. Avšak užitím algoritmu dělení, je jednoduché ukázat, že $S(x^3 + y, z^4) = yz^4 = (z^2 - z^2 + z - 1)(yz + y) + y$ takže $\overline{S(x^3 + y, z^4)}^G = y \neq 0$.

Definice 64. Nechť $F = (f_1, ..., f_s)$. Spřažení hlavních členů $LT(f_1), ..., LT(f_s)$ je s-tice polynomů $S = (h_1, ..., h_s) \in (k[x_1, ..., x_n])^s$ taková, že $\sum_{i=1}^{s} h_i LT(f_i) = 0$. Nechť $S(F)$ je podmnožina $(k[x_1, ..., x_n])^s$ obsahující všechna spřažení hlavních členů F.

Například pro $F = (x, x^2 + z, y + z)$ definuje trojice $S = (-x + y, 1, -x)$ jedno možné spřažení ze $S(F)$, jelikož platí $(-x + y)LT(x) + 1LT(x^2 + z) + (-x)LT(y + z) = 0$

Nechť $e_i = (0, ..., 0, 1, 0, ..., 0)$ jsou vektory, které mají jedničku na i-tém místě. Potom spřažení $S \in S(F)$ můžeme zapsat ve tvaru $S = \sum_{i=1}^{s} h_i e_i$. Jako příklad můžeme uvažovat spřažení pro S-polynomy. Pro každou dvojici $\{f_i, f_j\} \subset F$, kde $i < j$ a x^γ je nejmenší společný násobek hlavních členů polynomů f_i, f_j. Označme $S_{ij} = \frac{x^\gamma}{LT(f_i)} e_i - \frac{x^\gamma}{LT(f_j)} e_j$. Potom S_{ij} patří do $S(F)$. Jelikož $S(F)$ má konečnou bázi, každé $S \in S(F)$ můžeme vyjádřit jako lineární kombinaci bázových spřažení s polynomickými koeficienty.

Definice 65. Prvek $S \in S(F)$ je homogenní stupně α, kde $\alpha \in \mathbb{N}_0^n$, pokud $S = (c_1 x^{\alpha(1)}, ..., c_s x^{\alpha(s)})$, kde $c_i \in k$ a $\alpha(i) + multideg(f_i) = \alpha$ pro i taková, že $c_i \neq 0$.

Lemma 66. *Každý prvek $S(F)$ lze vyjádřit jednoznačně jako sumu homogenních prvků z $S(F)$.*

Tvrzení 67. *Dáno $F = (f_1, ..., f_s)$, každé spřažení $S \in S(F)$ můžeme zapsat jako $S = \sum_{i<j} u_{ij} S_{ij}$, kde $u_{ij} \in k[x_1, ..., x_n]$ a spřažení S_{ij} je definováno jako dříve.*

Příklad: $F = (x^2y^2 + z, xy^2 - y, x^2y + yz)$, použijeme lexikografické uspořádání s $x > y > z$. Dostaneme $S_{12} = (1, -x, 0), S_{13} = (1, 0, -y), S_{23} = (0, x, -y)$. Je vidět, že $S_{23} = S_{13} - S_{12}$. Spřažení S_{23} je tedy nadbytečné, jelikož ho můžeme získat jako lineární kombinaci S_{12} a S_{13}. Bázi spřažení tedy tvoří $\{S_{12}, S_{13}\}$.

Věta 68. *Báze $G = (g_1, ..., g_t)$ ideálu $\mathfrak{i}$ je Gröbnerova báze právě tehdy, když pro každý prvek $S = (h_1, ..., h_t)$ v homogenní bázi pro spřažení $S(G)$, máme $SG = \sum_{i=1}^{t} h_i g_i \underset{G}{\rightarrow} 0$.*

Tvrzení 69. *Je dáno $G = (g_1, ..., g_t)$, předpokládejme, že $S \subset \{S_{ij}; 1 \leq i < j \leq t\}$ je báze $S(G)$. Navíc předpokládejme, že máme různé prvky $g_i, g_j, g_k \in G$ takové, že $LT(g_k)$ dělí $LCM(LT(g_i), LT(g_j))$. Jestliže $S_{ik}, S_{jk} \in S$, pak $S - \{S_{ij}\}$ je také báze $S(G)$.*

Zavedeme značení $[i, j] \begin{cases} (i, j) \text{ pro } i < j \\ (j, i) \text{ pro } i > j \end{cases}$

Věta 70. *Nechť* $\mathfrak{i} = \langle f_1, ..., f_s \rangle$ *je polynomický ideál. Pak Gröbnerova báze pro* $\mathfrak{i}$ *může být zkonstruována v konečně mnoha krocích dle následujícího algoritmu*
INPUT: $F = \langle f_1, ..., f_s \rangle$
OUTPUT: Gröbnerova báze $G = (g_1, ..., g_t)$ *ideálu* $\mathfrak{i} = \langle f_1, ..., f_s \rangle$
$\{inicializace\}\ B := \{(i,j); 1 \leq i < j \leq s\}$
$G := F$
$t := s$
$\{iterace\}$ *WHILE* $B \neq 0$ *DO*
Vyber $(i,j) \in B$
IF $LCM(LT(f_i), LT(f_j)) \neq LT(f_i)LT(f_j)$ *AND NOT*
Test (f_i, f_j, B) *THEN*
$S := \overline{S(f_i, f_j)}^G$
IF $S \neq 0$ *THEN*
$t := t + 1, f_t := S$
$G := G \cup \{f_t\}$
$B := B \cup \{(i,t); 1 \leq i \leq t-1\}$
$B := B - \{(i,j)\}$,
kde Test (f_i, f_j, B) *nabývá hodnoty TRUE právě tehdy, když existuje* $k \notin \{i,j\}$ *tak, že* $[i,k]$ *a* $[j,k]$ *nejsou v* B *a současně* $LT(f_k)$ *dělí* $LCM(LT(f_i), LT(f_j))$.

3.2.2. F4 algoritmus

F4 algoritmus se používá pro výpočet Gröbnerových bází. Nahrazuje klasickou polynomickou redukci z Buchbergerova algoritmu souběžnou redukcí několika polynomů. Tento algoritmus je funkční pro všechna přípustná uspořádání, ale je vhodný především pro stupňované inverzní lexikografické uspořádání.

Matematické značení

Nechť $R[\mathbf{x}] = R[x_1, ..., x_n]$ je okruh polynomů. Polynom v n neurčitých nad okruhem R je definován jako zobrazení $f : \mathbb{N}_0^n \to \mathbb{R}$ s konečným nosičem (monom je restrikcí polynomu a má jednoprvkový nosič). Prvky $supp\ f \subseteq \mathbb{N}_0^n$ jsou nazývány členy f a jejich obrazy v $\mathbb{R}$ jsou nazývány koeficienty f. Označme $M(x_1, ..., x_n)$ nebo zkráceně M, množinu všech monomů v těchto proměnných. Zvolíme monomické uspořádání $<$ na M. Je-li $m = x_1^{\alpha_1}...x_n^{\alpha_n} \in M$, pak maximální stupeň m je definován jako $multideg(m) = \sum_{i=1}^n \alpha_i$. Nechť $f \in R[\mathbf{x}], f \neq 0$, pak $T(f)$ označíme množinu členů f. Maximální stupeň $f \neq 0$ je definován $multideg(f) = max\{multideg(m); m \in M(f)\}$. Definujeme hlavní člen $LT(f)$, hlavní monom $LM(f)$, hlavní koeficient $LC(f)$ s ohledem na uspořádání následovně: $LT(f) = max(T(f)), LM(f) = max(M(f))$, $LC(f) =$ koeficient u $LT(f)$. Je-li F podmnožina $R[\mathbf{x}]$, pak můžeme definici rozšířit: $LT(F) = \{LT(f); f \in F\}$, $LM(F) = \{LM(f); f \in F\}$, $M(F) = \{M(f); f \in F\}$. $\langle F \rangle$ označuje ideál generovaný F.

Nechť $f, g, p \in R[\mathbf{x}]$, kde $p \neq 0$ a nechť F je konečná podmnožina $R[\mathbf{x}]$, pak řekneme, že:

- f *se redukuje na* g *modulo* p $(f \equiv g (mod\ p))$, jestliže $\exists m \in M(f), \exists s \in M$ takové, že $s.LM(p) = m, g = f - \frac{a}{LC(p)} s.p$, kde a je koeficient u m v f

- f *se redukuje na* g *modulo* P ($f \equiv g(mod\ P)$), jestliže $f \equiv g(mod\ p)$ pro nějaké $p \in P$
- f je *redukovatelné modulo* p, jestliže existuje $g \in R[\mathbf{x}]$ tak, že $f \equiv g(mod\ p)$
- f je *redukovatelné modulo* P, jestliže existuje $g \in R[\mathbf{x}]$ tak, že $f \equiv g(mod\ P)$
- f je *top-redukovatelné modulo* P, jestliže existuje $g \in R[\mathbf{x}]$ tak, že $f \equiv g(mod\ P)$ a $LM(g) < LM(f)$
- $f \stackrel{*}{=} g(mod\ P)$ je reflexivní a tranzitivní uzávěr $f \equiv g(mod\ P)$
- $S-polynom$ f a g je definován:

$$S(f,g) = LC(g)\frac{LCM(LM(f), LM(g))}{LM(f)}f - LC(f)\frac{LCM(LM(f), LM(g))}{LM(g)}g$$

Příklad 1:
$f = 2x^4 + x^2 + 2x + 1 \Rightarrow M(f) = \{x^4, x^2, x, 1\}$
$p = x^2 + 1 \Rightarrow LM(p) = x^2, LC(p) = 1$
f se redukuje např. na $-x^2 + 2x + 1$, protože: $m = x^4, s = x^2, a = 2$, což zaručí splnění podmínek
$x^2.x^2 = x^4$
$-x^2 + 2x + 1 = 2x^4 + x^2 + 2x + 1 - \frac{2}{1}x^2\,(x^2 + 1)$

Najděte f, p tak, aby f byl *redukovatelný modulo* p, ale nebyl *top-redukovatelný modulo* p.
Příklad 2:
$f = x^5 + x^3 + 2x + 2 \Rightarrow M(f) = \{x^5, x^3, x, 1\}$
$p = x \Rightarrow LM(p) = x, LC(p) = 1$
$s.LM(p) = m$
$1.x = x \Rightarrow a = 2$
$g = x^5 + x^3 + 2x + 2 - \frac{2}{1}x = x^5 + x^3 + 2$

Vícerozměrné příklady:
Příklad 3:
$f = 2x^2y^2z^2 - 2x^2yz^2 + 3x^2yz + 4xyz - 2xy + 4xz + 5x + 2y + 3z + 2 \Rightarrow$
$M(f) = \{x^2y^2z^2, x^2yz^2, x^2yz, xyz, xy, xz, x, y, z, 1\}$
$p = 2xyz + xy + z + 2 \Rightarrow LM(p) = xyz, LC(p) = 2$
$s.LM(p) = m$
$xyz.xyz = x^2y^2z^2 \Rightarrow a = 2$
$g = 2x^2y^2z^2 - 2x^2yz^2 + 3x^2yz + 4xyz - 2xy + 4xz + 5x + 2y + 3z + 2 - \frac{2}{2}xyz.(2xyz + xy + z + 2) =$
$-x^2y^2z - 2x^2yz^2 + 3x^2yz - xyz^2 + 2xyz - 2xy + 4xz + 5x + 2y + 3z + 2$
Příklad 4:
$f = 2x^2y^2z^2 - 2x^2yz^2 + 3x^2yz + 4xyz - 2xy + 4xz + 5x + 2y + 3z + 2 \Rightarrow$
$M(f) = \{x^2y^2z^2, x^2yz^2, x^2yz, xyz, xy, xz, x, y, z, 1\}$
$p = 2xyz + xy + z + 2 \Rightarrow LM(p) = xyz, LC(p) = 2$
$s.LM(p) = m$
$1.xyz = xyz \Rightarrow a = 4$
$g = 2x^2y^2z^2 - 2x^2yz^2 + 3x^2yz + 4xyz - 2xy + 4xz + 5x + 2y + 3z + 2 - \frac{4}{2}(2xyz + xy + z + 2) =$
$2x^2y^2z^2 - 2x^2yz^2 + 3x^2yz - 4xy + 4xz + 5x + 2y + z - 2$

Definice 71. Mějme matici M typu $s \times m$. Označme m_{ij} prvek v i-tém řádku a j-tém sloupci matice M. Mějme monomy $m_1, ..., m_m$ uspořádané lexikograficky, kde $m_1 > m_2 > \cdots > m_m$. Dále označíme $M_M = [m_1, ..., m_m]$. Nechť $\underbrace{R^m = R \oplus \cdots \oplus R}_{m-krat}$ je R-modulem, pak $(\epsilon_i)_{i=1,...,m}$ je kanonická báze R^m. Uvažujme lineární zobrazení $\varphi_{M_M} : V_{M_M} \to R^m$ (kde V_{M_M} je podmodul $R[\mathbf{x}]$ generovaný aditivně V_{M_M}) takové, že $\varphi_{M_M}(m_i) = \epsilon_i$. Inverzní funkci označíme ψ_{M_M}. Aplikace ψ_{M_M} umožňuje interpretovat vektory R^m jako polynomy. Označíme (M, M_M) matici s interpretací.

Definice 72. Nechť (M, M_M) je matice typu $s \times m$ s interpretací, pak zkonstruujeme množinu polynomů

$$radky(M, M_M) := \{\psi_{M_M}(radek(M, i); i = 1, ..., s\} - \{0\},$$

kde $radek(M, i)$ je i-tý řádek M (prvek R^m). Opačně nechť $l = l_1, ..., l_s$ je seznam polynomů a $M_l = [m_1, ..., m_m]$ je množina monomů uspořádána lexikograficky, kde $m_1 > m_2 > \cdots > m_m$. Nyní zkonstruujeme matici A typu $s \times m$ ($s = vel(l), m = vel(M_l)$ tak, aby

$$a_{ij} := koef(l_i, M_l; i = 1, ..., s; j = 1, ..., m).$$

Matici (a_{ij}) označíme $A^{(l,M_l)}$.

Definice 73. Mějme matici s interpretací (M, M_M). Označíme $F = radky(M, M_M)$. Spočítáme $\widetilde{F}$ jako redukovanou Gröbnerovu bázi F pro lexikografické uspořádání $m_1 > m_2 > \cdots > m_m$. Z této báze můžeme rekonstruovat matici $\widetilde{M} = A^{(\widetilde{F}, M_M)}$. $\widetilde{M}$ nazveme jediný řádkově schodovitý tvar matice M s ohledem na lexikografické uspořádání $m_1 > m_2 > \cdots > m_m$. Také můžeme říci, že $\widetilde{F}$ je řádkově schodovitá báze F s ohledem na lexikografické uspořádání $m_1 > m_2 > \cdots > m_m$.

F4 algoritmus

Víme, že při výpočtu Buchbergerova algoritmu máme více voleb:

- vybrat kritický pár ze seznamu kritických párů
- vybrat jeden reduktor ze seznamu reduktorů

Buchberger dokázal, že tyto volby nemají vliv na správnost algoritmu, ale je známo, že jsou zásadní pro celkový čas výpočtu. Nejlepší strategií je vyšetřovat pouze hlavní monomy polynomů k vybrání volby. Uvažujme, že všechny vstupní polynomy mají stejné hlavní monomy. V tomto případě jsou všechny kritické páry shodné a není možné o volbě rozhodnout. Tento problém vyřešíme jednoduše. Neuděláme žádnou volbu. Přesněji, místo volby jednoho kritického páru v každém kroku vybereme podmnožinu kritických párů najednou.

Definice 74. *Kritický pár* polynomů (f_i, f_j) je prvek $M^2 \times R[\mathbf{x}] \times M \times R[\mathbf{x}]$,

$$Par(f_i, f_j) := (LCM_{ij}, m_i, f_i, m_j, f_j)$$

takové, že $LCM(Par(f_i, f_j)) = LCM_{ij} = LM(m_i f_i) = LM(m_j f_j) = LCM(LM(f_i), LM(f_j))$.

Definice 75. *Stupeň kritického páru* $p_{i,j} = Par(f_i, f_j)$, $deg(p_{i,j})$ je $deg(LCM_{i,j})$. Definujeme dvě projekce $Leva(p_{i,j}) := (m_i, f_i), Prava(p_{i,j}) := (m_j, f_j)$. Pokud $(m, p) \in M \times R[\mathbf{x}]$, pak označíme $mult((m, p))$ vyhodnocení součinu $m \times p$.

Algoritmus převzat z [2].

INPUT: $\begin{cases} \text{F- konečná podmnožina } R[\mathbf{x}] \\ \text{Sel- funkce: Seznam(Paru)} \rightarrow \text{Seznam(Paru) tak, že } Sel(l) \neq 0, \text{ jestliže } l \neq 0 \end{cases}$
OUTPUT: konečná podmnožina $R[\mathbf{x}]$
$G := F, \widetilde{F}_0^+ := F, d := 0$
$P := \{Par(f, g); f, g \in G, f \neq g\}$
WHILE $P \neq 0$ DO
$d := d + 1$
$P_d := Sel(P)$
$P := P \backslash P_d$
$L_d := Leva(P_d) \cup Prava(P_d)$
$\widetilde{F}_d^+ := Redukce(L_d, G)$
FOR $h \in \widetilde{F}_d^+$ DO
$P := P \cup \{Par(h, g); g \in G\}$
$G := G \cup \{h\}$
RETURN G

Redukce

INPUT: $\begin{cases} \text{L- konečná podmnožina } M \times R[\mathbf{x}] \\ \text{G- konečná podmnožina } R[\mathbf{x}] \end{cases}$
OUTPUT: konečná podmnožina $R[\mathbf{x}]$ (možné také prázdná množina)
$F :=$Symbolicke-prepocesovani(L, G)
$\widetilde{F} :=$ redukce na řádkově schodovitý tvar z F s ohledem na uspořádání
$\widetilde{F}^+ := \{f \in \widetilde{F}; LM(f) \notin LM(F)\}$
RETURN $\widetilde{F}^+$

Pozn. : Ekvivalentní definice $\widetilde{F}^+$ je $\widetilde{F}^+ := \{f \in \widetilde{F}; f \text{ } top \text{ } ireducibilni \text{ } G\}$
Nyní popíšeme hlavní funkci tohoto algoritmu, tedy konstrukci matice F. na tento subalgoritmus se můžeme dívat jako na klasickou redukci všech uvažovaných polynomů, pokud nahradíme standartní aritmetiku následovně: nechť $0 \neq x, 0 \neq y \in R$, pak $x + y = 1, x * y = 1, x * 0 = 0, x + 0 = 1$. Tedy jedná se opravdu o symbolické přeprocesování.

Symbolicke-prepocesovani

INPUT: $\begin{cases} \text{L- konečná podmnožina } M \times R[\mathbf{x}] \\ \text{G- konečná podmnožina } R[\mathbf{x}] \end{cases}$
OUTPUT: konečná podmnožina $R[\mathbf{x}]$
$F := \{m \times f; (m, f) \in L\}$
$Done := LM(F)$
WHILE $M(F) \neq Done$ DO
$m \in M(F) \backslash Done$

$Done := Done \cup \{m\}$
IF m je top reducibilní modulo G THEN
$m := \acute{m}LM(F)$ pro nějaké $f \in G, \acute{m} \in M$
$F := F \cup \{\acute{m}f\}$
RETURN F

Pozn. : Jestliže $size(Sel(l)) = 1$ pro všechna $l \neq 0$, pak F4 algoritmus je Buchbergerovým algoritmem. V tomto případě funkce Sel odpovídá strategii výběru v Buchbergerově algoritmu.

3.2.3. Čuang-c'ův algoritmus

Algoritmus pro výpočet polynomických rovnic více proměnných nad konečnými poli. Nejdříve se v této kapitole seznámíme se samotným algoritmem a ilustrujeme ho na příkladech.

Algebraické pozadí

Nechť k je pole s q prvky. Mějme m polynomů $f_0, ..., f_{n-1} \in k[x_0, ..., x_{n-1}]$. Naším cílem je najít všechny n-tice $(a_0, ..., a_{n-1}) \in k^n$ takové, že

$$\begin{aligned} f_0(a_0, ..., a_{n-1}) &= 0 \\ &\vdots \\ f_{n-1}(a_0, ..., a_{n-1}) &= 0 \end{aligned}$$

Pracujeme s polem, kde $x_i^q = x_i$. Klíčovou myšlenkou tohoto algoritmu je posunout se z prostoru polynomů $k[x_0, ..., x_{n-1}]$ s koeficienty v k do prostoru polynomu $K[X]$ s koeficienty ve vhodně vybraném rozšířeném poli K.

Pro názornost dále předpokládejme, že $m = n$. Vybereme ireducibilní polynom $g(y) \in k[y]$ stupně n. Pak $K = k[y]/(g(y))$ je rozšíření k stupně n. Nechť ϕ je k-lineární zobrazení takové, že ztotožňuje K s n-rozměrným vektorovým prostorem k^n, tj. $\phi : k^n \to K$ definované $\phi(a_0, ..., a_{n-1}) = a_0 + a_1 y + \cdots + a_{n-1} y^{n-1}$. Nechť $f : k^n \to k^n$ je polynomické zobrazení definované $f = (f_0, ..., f_{n-1})$. Převedeme f do rozšířeného pole K užitím ϕ a vytvořením zobrazení $F : K \to K$ definovaného $F = \phi \circ f \circ \phi^{-1}$. Užitím Lagrangeovy interpolační formule můžeme považovat F za polynom v $K[X]$. Opravdu F má jedinou reprezentaci v podílovém okruhu $K[X]/(X^{q^n} - X)$. Pro dané f dostaneme odpovídající F vyřešením soustavy lineárních rovnic. Následující teorém nám ukazuje přesný tvar této reprezentace.

Teorém 76. *Užitím notace zavedené výše, pro lineární polynomické zobrazení* $f = (f_0, ..., f_{n-1})$ *máme*

$$F(X) = \sum_{i=0}^{n-1} \beta_i X^{q^i} + \alpha \ mod \ (X^{q^n} - X),$$

pro nějaké $\beta_i, \alpha \in K$. *Jestliže* f *je kvadratické polynomické zobrazení, pak*

$$F(X) = \sum_{i=0}^{n-1} \sum_{j=i}^{n-1} \gamma_{ij} X^{q^i + q^j} + \beta_i X^{q^i} + \alpha \ mod \ (X^{q^n} - X),$$

pro nějaké $\gamma_{ij}, \beta_i, \alpha \in K$.

Nyní je zřejmé, že se můžeme volněji pohybovat mezi funkcemi s více proměnnými a funkcemi s jednou proměnnou. Máme-li systém rovnic, základní strategií je převést polynomické zobrazení f na F v rozšířeném poli K. Kořeny F, který je dán Teorémem 77, přesně odpovídají řešení původní soustavy rovnic definovaných nad k. Tedy máme-li kořeny v K, převedeme je do k^n pomocí ϕ^{-1}.

Zdůrazněme základní rozdíl mezi tímto algoritmem a ostatními algoritmy. Tento algoritmus lze použít pouze nad konečnými poli.

Algoritmus byl pojmenován po antickém čínském filosofovi Čuang-c'ovi. "Jednou se Čuang-c'ovi zdálo, že je motýl. Motýl poletující kolem. Byl šťastný a dělal, co se mu zachtělo. Nevěděl, že je Čuang-c'em. Náhle se probudil a byl opět pevným a nezaměnitelným Čuang-c'em. Nevěděl ale, zda to byl Čuang-c, který snil o tom, že je motýl, nebo motýl sní o tom, že je Čuang-c'em. Mezi Čuang-c'em a motýlem musí být přeci nějaký rozdíl."

Toto se nazývá transformace věcí.

Algoritmus

Algoritmus převzat z [4].

Začněme případem, kdy $m = n$, tedy kdy máme stejný počet rovnic a proměnných. Čuang-c'ův algoritmus má na vstupu polynomy $f_0, ..., f_{n-1} \in k[x_0, ..., x_{n-1}]$ a kladné číslo D. D je horní hranice stupně polynomických rovnic, které máme vyřešit. Výsledkem algoritmu je n-tice $(a_0, ..., a_{n-1}) \in k^n$ taková, že $f_i(a_0, ..., a_{n-1}) = 0$ pro $i = 0, ..., n-1$.

- **Krok 1:** Vybereme ireducibilní polynom $g(y) \in k[y]$ stupně n a definujeme $K = k[y]/(g(y))$. Nechť $\phi : k^n \to K$ je definováno jako v předchozím. Převedeme $f = (f_0, ..., f_{n-1})$ do K na $F = \phi \circ f \circ \phi^{-1}$ a spočteme polynomickou reprezentaci $F(X)$ modulo $x^{q^n} - X$. Je-li $deg(F(X)) \leq D$, pak přejdeme k poslednímu kroku, jinak postupujeme následovně.

- **Krok 2:** Nechť $G = Gal(K/k)$ je Galoisovo pole K nad k skládající se z Frobeniových zobrazení $G_i(X) = X^{q^i}$, pro $i = 0, ..., n-1$. Počítejme $F_i(X) = G_i \circ F(X) = F(X)^{q^i} \mod(X^{q^n} - X)$, pro $i = 0, ..., n-1$. Poznamenejme, že $F_0(X) = F(X)$. Existuje-li F_i takové, že $deg(F_i(X)) \leq D$, pak přejdeme na poslední krok, jinak postupujeme následovně.

- **Krok 3:** Nechť N je počet monomů, které se vyskytují v nějakém $F_i(X)$. Pro každé $F_i(X)$ vytvoříme řádkový vektor v K^N, kde vstupem jsou koeficienty z $F_i(X)$ seřazené v sestupném pořadí. Dále zkonstruujeme $n \times N$ matici užitím těchto vektorů. Použijeme Gaussovu eliminaci k získání nové množiny t bází polynomů $S = S_0(X), ..., S_{t-1}(X)$. Jinými slovy eliminujeme monomy nejprve nižších stupňů. Označíme prvky S tak, že $S_{t-1}(X)$ je prvek s nejnižším stupněm. Jestliže $deg(S_{t-1}(X)) \leq \leq D$, pak přejdeme na poslední krok, jinak postupujeme následovně.

- **Krok 4:** Musí platit, že polynom z S s nejmenším stupněm má stupeň větší než D. Pro každé $i = 0, ..., t-1$ a $j = 0, ..., n-1$ spočítáme $X^{q^j} S_i(X) \mod(X^{q^n} - X)$. Jako v předchozím použijeme Gaussovu eliminaci na matici sestavenou ze soustavy polynomů. Tím získáme novou bázi polynomů $\acute{S}$. Nechť $\acute{S}_{t-1}(X)$ je polynom

v $\acute{S}$ s nejnižším stupněm. Jestliže $deg(\acute{S}_{t-1}(X)) \leq D$, pak přejdeme k poslednímu kroku, jinak S nahradíme $\acute{S}$ a opakujeme tento krok.

- **Krok 5:** V tomto bodě máme polynom $G(X)$, kde $deg(G(X)) \leq D$. Vyřešíme $G(X) = 0$ pomocí vhodně vybraného rychlého řešiče polynomických rovnic či faktorizačním algoritmem a tím obdržíme $W = \{\alpha \in K; G(\alpha) = 0\}$. Řešení $F(X) = 0$ je podmnožina $\{\alpha \in W; F(\alpha) = 0\}$.

Pozn. 1: Čuang-c'ův algoritmus můžeme modifikovat v případě, kdy máme méně rovnic než proměnných, tedy když $m < n$. Nechť $\pi : k^m \to k^n$ je injektivní zobrazení definované $\pi(a_0, ..., a_{m-1}) = (a_0, ..., a_{m-1}, 0, ..., 0)$. $f = (f_0, ..., f_{m-1})$ nahradíme $\pi \circ f$ a dále postupujeme jako v předchozím. Musíme poznamenat, že pokud je m příliš malé, pak existuje velký počet řešení (přesněji q^{n-m-1}) a navíc polynomy v ideálu generovaném $F_i(X)$ budou velkého stupně.

Pozn. 2: Pokud máme více rovnic než proměnných, tedy $m > n$, pak musíme Čuang-c'ův algoritmus opět modifikovat. Předpokládejme, že $m = nr + l$, pro nějaké $0 \leq l < n$ a rozdělme $f_0, ..., f_{m-1}$ na r množin po n polynomech a jednu množinu o l polynomech. Je-li m násobek n, pak převedeme všech r množin polynomů na polynom v $K[X]$ jako v předchozím. Jestliže $l \neq 0$, převedeme poslední množinu l polynomů.

Příklady

Příklad 1

Nechť $k = GF(2^2)$. Definujeme polynomické zobrazení $f : k^2 \to k^2$. Jeho složky jsou

$$f_0(x_0, x_1) = x_0^2 + x_1 + 1$$

$$f_1(x_0, x_1) = x_1^2 + x_0 x_1 + 1$$

v $k[x_0, x_1]$. Tabulka pro sčítání a násobení v $GF(4)$

+	0	1	2	3
0	0	1	2	3
1	1	0	3	2
2	2	3	0	1
3	3	2	1	0

*	0	1	2	3
0	0	0	0	0
1	0	1	2	3
2	0	2	3	1
3	0	3	1	2

Vybereme ireducibilní polynom stupně 2 s koeficienty v k: $g(y) = y^2 + y + 3$.
Zobrazení $\phi : k^2 \to K$ je definováno následovně

$$\phi(x_0, x_1) = x_0 + x_1 y = X$$

Řešení:

a)Použití Teorému 77: Jestliže $f = (f_0, f_1, ..., f_{n-1})$ je kvadratické polynomické zobrazení, pak

$$F(X) = \sum_{i=0}^{n-1} \sum_{j=i}^{n-1} \gamma_{ij} X^{q^i + q^j} + \sum_{i=0}^{n-1} \beta_i X^{q^i} + \alpha \ mod \ (X^{q^n} - X),$$

kde $\gamma_{ij}, \beta_i, \alpha \in K$. Dle tohoto teorému máme

$$F(X) = \gamma_{11}X^8 + \gamma_{01}X^5 + \beta_1 X^4 + \gamma_{00}X^2 + \beta_0 X + \alpha$$

Nyní již zbývá dopočítat pouze koeficienty. Víme, že

$$F(x_0 + x_1 y) = x_0^2 + x_1 + 1 + (x_1^2 + x_0 x_1 + 1)y$$

tedy např.

$$7 = y + 3 \Rightarrow F(3 + 1y) = 3^2 + 1 + 1 + (1^2 + 3.1 + 1)y = 2 + 3y = 14.$$

Vypíšeme potřebný počet bodů

$$0 \mapsto 5$$
$$1 \mapsto 4$$
$$2 \mapsto 6$$
$$3 \mapsto 7$$
$$4 \mapsto 0$$
$$5 \mapsto 5$$
$$6 \mapsto 11$$
$$7 \mapsto 14$$
$$8 \mapsto 11$$

Nyní dosadíme tyto hodnoty do dané rovnice a vyřešíme soustavu rovnic. Tím získáme výsledné polynomické zobrazení

$$F(X) = 4X^8 + 4X^5 + X^4 + X^2 + X + 5.$$

b)Přímý výpočet Lagrangeova interpolačního polynomu

Lagrangeův interpolační polynom je dán vztahem

$$L_n(x) = \sum_{k=0}^{n} y_k l_{n,k}(x), \qquad l_{n,k}(x) = \prod_{k=0, k\neq n}^{n} \frac{X - x_k}{x_n - x_k}$$

Nejdříve potřebujeme sestavit tabulku hodnot

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
y	5	4	6	7	0	5	11	14	11	2	4	13	14	3	9	4

Nyní již můžeme dosadit do daného vzorce. Např.

$$l_{15,6} = \frac{(X-0)(X-1)\cdots(X-5)(X-7)\cdots(X-15)}{(6-0)(6-1)\cdots(6-5)(6-7)\cdots(6-15)}$$

Z důvodu rozšíření pole na $GF(16)$ musíme sčítat i násobit v tomto poli. Při násobení odečítáme daný ireducibilní polynom, tedy např. $11.12 = (2y+3)3y = y^2 + 2y$ a po odečtení $y^2 + y + 3$ dostáváme výsledek $3y + 3 = 15$.

Stejným způsobem pokračujeme ve výpočtech dále až k výslednému Lagrangeovu interpolačnímu polynomu

$$L(X) = 4X^8 + 4X^5 + X^4 + X^2 + X + 5.$$

Příklad 2

Nechť $k = GF(2^2)$. Definujeme polynomické zobrazení $f : k^2 \to k^2$. Jeho složky jsou

$$f_0(x_0, x_1) = 2x_0^3 + 3x_1^2 x_0 + 1$$

$$f_1(x_0, x_1) = x_1^3 + 2x_0^2 x_1 + 2$$

v $k[x_0, x_1]$. Mějme stejný ireducibilní polynom jako v předchozím příkladě.

Řešení:

a)Použití Teorému 77: Nyní je $f = (f_0, f_1, ..., f_{n-1})$ již kubické polynomické zobrazení, tedy

$$F(X) = \sum_{i=0}^{n-1} \sum_{j=i}^{n-1} \sum_{k=j}^{n-1} \delta_{ijk} X^{q^i+q^j+q^k} + \sum_{i=0}^{n-1} \sum_{j=i}^{n-1} \gamma_{ij} X^{q^i+q^j} + \sum_{i=0}^{n-1} \beta_i X^{q^i} + \alpha \ mod \ (X^{q^n} - X),$$

kde $\gamma_{ij}, \beta_i, \alpha \in K$. Tedy po přeznačení koeficientů máme

$$F(X) = aX^{12} + bX^9 + cX^8 + dX^6 + eX^5 + fX^4 + gX^3 + hX^2 + iX + j$$

Nyní již zbývá dopočítat pouze koeficienty. Víme, že

$$F(x_0 + x_1 y) = 2x_0^3 + 3x_1^2 x_0 + 1 + (x_1^3 + 2x_0^2 x_1 + 2)y$$

Nyní musíme sestavit tabulku hodnot

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
y	9	11	11	11	13	4	10	1	13	1	4	10	13	10	1	4

Tyto hodnoty dosadíme do polynomické rovnice. Tím získáme soustavu lineární rovnic, kterou vyřešíme.

Výsledné polynomické zobrazení je pak dáno vztahem

$$F(X) = 8X^{12} + 2X^9 + 8X^6 + 9.$$

b)Přímý výpočet Lagrangeova interpolačního polynomu Zde je postup naprosto totožný jako u předchozího příkladu, pouze použijeme příslušnou tabulku hodnot. Výsledný Lagrangeův interpolační polynom je tedy dán vztahem

$$L(X) = 8X^{12} + 2X^9 + 8X^6 + 9.$$

Závěr

Poznamenejme, že Čuang-c'ův algoritmus není novým algoritmem, ale novým způsobem, jak nahlížet na problém řešit polynomické rovnice ve více proměnných nad konečnými poli. Náš problém převedeme na rozšířené pole, kde se z něj stává problém o jedné proměnné. Poté použijeme již existující vhodné postupy pro řešení polynomické rovnice v jedné proměnné nad konečným polem. Tedy vlastně ztotožňujeme vícerozměrný a jednorozměrný případ. To také znamená, že někdy může být prospěšné se na jednorozměrný případ nad daným konečným polem dívat jako na množinu vícerozměrných polynomických rovnic nad menším konečným polem.

3.3. Ataky

Tato kapitola je inspirována [3], [5].

3.3.1. Patarinův atak

Multivariační kryptosystémy s veřejným klíčem

Bezpečnost multivariačních kryptosystémů s veřejným klíčem (MPKC) spoléhá na náročnost řešení systému nelineárních polynomických rovnic s mnoha proměnnými. Veřejný klíč MPKC je většinou množina kvadratických polynomů, které jsou odvozeny ze skládání zobrazení.

Krátký popis Matsumoto-Imai algoritmu

Matematické vlastnosti

Nechť k je konečné pole charakteristiky 2 a nechť $q = 2^m$ je počet prvků k. Obecný tvar MPKC je tvaru $Y = (y_1, ..., y_m) = F(x_1, ..., x_n) = J_2 \circ \phi \circ J_1(x_1, ..., x_n)$, kde $F : k^n \to k^m; J_1 : k^n \to k^n, J_2 : k^m \to k^m$ jsou afinní transformace. Nechť L_n je rozšíření k stupně N a nechť θ je celé číslo. Pak funkce $f : L_N \to L_N : x \mapsto x^{1+2^{m\theta}}$ je bijekce, jestliže $1 + 2^{m\theta}$ je nesoudělné s $2^{mN} - 1$. Tedy je-li f bijekce, lze jednoduše invertovat a existuje inverzní funkce f^{-1} taková, že $f^{-1}(x) = x^{\hbar}$, kde $\hbar$ je multiplikativní inverze $1 + 2^{m\theta}$ modulo $2^{mN} - 1$. Zobrazení $\pi : k \to L_N$ je N- lineární izomorfismus. Zobrazení ϕ nazýváme centrální zobrazení, které je definováno $\phi = \pi \circ \tilde{\phi} \circ \pi^{-1}$. Zobrazení $\tilde{\phi}$ nad L_N je rozdílné pro různé MPKC. Jedna z hlavních myšlenek, jak zkonstruovat tento systém byla započata Matsumotem a Imaiem.

Popis Matsumoto-Imai algoritmu

Pole k s 2^m prvky je veřejné. Každá zpráva má nm bitů, kde n je další veřejné celé číslo. n je rozděleno následovně $n = n_1 + \cdots + n_d$, kde $n_1, ..., n_d$ jsou opět celá čísla. Poté vytvoříme d rozšíření $k, L_{n_1}, ..., L_{n_d}$ stupně $n_1, ..., n_d$. Hodnotu reprezentovanou prvky k nazveme slovo. Např. prvek $L_{n_e}; 1 \leq e \leq d$, může být slovo délky n_e. Použijeme kvadratické funkce $f_1, ..., f_d$ dávající d slov. Těchto d slov pak překombinujeme na slovo délky n.

Tajné položky jsou:

1. Dvě afinní bijekce $s, t : k^n \to k^n$ (mohou být reprezentovány bází polynomů celkového stupně 1 a koeficienty polynomů jsou prvky k).

2. Dělení n na d celých čísel $n = n_1 + \cdots + n_d$.

3. Reprezentace polí $L_{n_1}, ..., L_{n_d}$. Tyto reprezentace jsou dány volbou d ireducibilních polynomů. Označme ψ_l izomorfismus z k^{n_l} do L_{n_e} daný těmito reprezentacemi, $1 \leq \leq e \leq d$.

4. Celá čísla $\theta_1, ..., \theta_d$ taková, že $1 \leq \theta_e \leq n_e$ a $GCD(2^{\theta_e} + 1, 2^{mn_e} - 1) = 1, 1 \leq e \leq d$. Tato celá čísla θ_e udávají kvadratické funkce $f_1, ..., f_d$.

Důležitý bod je, že skládání všech těchto operací je kvadratická funkce s prvky v bázi. Takže tato funkce může být dána n polynomy nad $k, (y_1, ..., y_n)$ (tyto polynomy dávají šifrovaný text y z původního textu x).

Veřejné položky jsou:

1. Pole k délky 2^m, délka zpráv n.
2. n polynomů $(y_1, ..., y_n)$ v n proměnných nad k.

Tudíž každý může zašifrovat zprávu.

Značení

m je stupeň pole k, n je počet prvků v k v každé zprávě, d je počet celých čísel v tajném dělení $n : n = n_1 + \cdots + n_d$. Nechť e je index, $1 \leq e \leq d$ a x je původní text a y šifrovaný text. Označme L_{n_e} rozšíření k stupně n_e. a_e je prvek L_{n_e} afinní v x, b_e je prvek L_{n_e} afinní v y, θ_e je tajný parametr takový, že

$$b_e = a_e^{1+2^{m\theta_e}} \tag{3.13}$$

Navíc pro jednoduchost označíme $\theta_e = \theta, a_e = a, b_e = b$.

Skupina slabých klíčů

Ukážeme, že existuje mnoho slabých klíčů v algoritmu MI. Je však velmi lehké se jim vyhnout a tudíž to nepovažujeme za závažný problém tohoto algoritmu.

Rovnici (3.13) můžeme zapsat ve tvaru $a = b^{\not{h}_e}$, kde $\not{h}_e$ je multiplikativní inverze $1 + 2^{m\theta_e}$ modulo $2^{mn_e} - 1$.

Nechť α je celé číslo, označíme $HW(\alpha)$ počet 1 ve výrazu α v bázi 2 (HW zastupuje Haminngovu váhu v bázi 2). Nechť x_i je bit x a y_i je bit $y (1 \leq i \leq mn)$. Každá hodnota $y_j, 1 \leq j \leq n$ má kvadratický výraz v hodnotě $x_i, 1 \leq i \leq n$. Jednoduše každá hodnota x_j má výraz jako polynom stupně $sup_{e,1\leq e\leq d} HW(\not{h}_e)$ v hodnotách y_i. Je dobré, když $HW(\not{h}_e)$ není příliš malé pro neznámé e. Předpokládejme, že nemáme tento případ, tedy pro jednu neznámou $e, 1 \leq e \leq d$ je $HW(\not{h}_e)$ velmi malé a

$$a = b^{\not{h}_e} \tag{3.14}$$

Jestliže a je afinní v x, existují hodnoty $\alpha_{1i}, 0 \leq i \leq mn$ takové, že $a_1 = \alpha_{10} + \sum_{i=1}^{mn} \alpha_{1i} x_i$.

Z (3.14) víme, že a_1 má polynomiální výraz celkového stupně $HW(\not{h}_e)$ v $b_1, ..., b_{n_e m}$. Tedy všechny hodnoty $b_1, ..., b_{n_e m}$ jsou afinní v $y_1, ..., y_{nm}$, tudíž a_1 má polynomický výraz celkového stupně $HW(\not{h}_e)$ v $y_1, ..., y_{nm}$. Takže máme polynom P celkového stupně $HW(\not{h}_e)$ tak, že

$$\alpha_{10} + \sum_{i=1}^{mn} \alpha_{1i} x_i = P(y_1, ..., y_{mn}) \tag{3.15}$$

Podobně pro $a_2, a_3, ..., a_{n_e m}$. Tedy máme nejméně $n_e m$ rovnic podobných (3.15), stupně 1 v x_i a celkového stupně $HW(\not{h}_e)$ v y_i. Takže jestliže pro určité $e, HW(\not{h}_e)$ je velmi malé (např. ≤ 4), chceme najít $n_e m$ rovnic podobných (3.15) (a to i dokonce když existuje f tak,

že $HW(\not{h}_e)$ je velmi velké). Při hledání těchto rovnic budeme jednoduše psát nejobecnější tvar rovnic stupně $HW(\not{h}_e)$. Generováním nějakých hodnot pro x a y z veřejného tvaru obdržíme rovnice stupně 1 s koeficienty polynomů.

Po nasbírání dostatečného počtu rovnic, Gaussovou redukcí na tyto rovnice, najdeme vektorový prostor řešení pro koeficienty polynomů. Tudíž najdeme nejméně $n_e m$ nezávislých rovnic podobných (3.15). Nyní z těchto rovnic, když je dáno y, máme okamžitě $n_e m$ rovnic stupně 1 bitů x_i původního textu. To může být velmi nebezpečné.

Závěr: Všechny hodnoty n_e, θ_e musí být vybrány tak, aby řád $HW(\not{h}_e)$ nebyl příliš malý (např. ≥ 6).

První obecný útok na všechny klíče

Příklad

Předpokládejme, že $m = 1, \theta = 1$.

$$b = a^3 \tag{3.16}$$

Nechť $(b_1, ..., b_{n_e})$ je reprezentace b v L_{n_e} a nechť $(a_1, ..., a_{n_e})$ je reprezentace a v L_{n_e}. Z (3.16) vidíme, že $\forall b_j, 1 \leq j \leq n_e$ máme kvadratický výraz v $(a_1, ..., a_{n_e})$, protože $b = aa^2$, a^2 je lineární (protože $m = 1$). Avšak rádi bychom našli výraz, který dává hodnoty a_j z b_j (namísto b_j z a_j). První myšlenka je samozřejmě napsat

$$a = b^{\not{h}} \tag{3.17}$$

ale v nejvíce případech $HW(\not{h})$ je velké, tudíž (3.17) dává úporný výraz pro hodnoty a_j.

Začněme opět z (3.16) a vynásobme oba výrazy z (3.16) a. Obdržíme

$$ba = a^4 \tag{3.18}$$

Rovnice (3.18) dává n_e rovnic stupně 1 na b_j hodnot, stupně 1 na a_j hodnot. (Protože a^4 je lineární v a, protože $m = 1$.) Navíc $\forall b \neq 0$ jsou zde právě 2 řešení (3.18): $a = 0, a = b^{\not{h}}$. Z rovnice (3.18) víme, že existuje rovnice tvaru

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\gamma_{ij}x_i y_j + \sum_{i=1}^{n}\alpha_i x_i + \sum_{i=1}^{n}\beta_i y_i + \delta_0 = 0 \tag{3.19}$$

Tyto rovnice platí $\forall x, y$, pak x je původní text k y. Navíc jestliže $b = 0$, existuje jenom jedno řešení pro a z (3.18).

Nutně musíme mít nejméně n_e formálně nezávislých rovnic (3.19). Avšak pro danou hodnotu y můžeme říci, že budeme mít nejméně $n_e - 1$ nezávislých rovnic (3.19) (a ne n_e), protože (3.18) má 2 řešení pro a, když $b \neq 0$.

Vybráním několika hodnot pro x a spočítáním hodnot y z x z veřejného tvaru, dále nahrazením těchto hodnot $x_i, y_i, 1 \leq i \leq n$ ve (3.19), obdržíme rovnice stupně 1 ve $n^2 + n + n + 1 = (n + 1)^2$ proměnných $\gamma_{ij}, \alpha_i, \beta_i, \delta_0$.

Rychle najdeme všechny rovnice (3.19) (Gaussovou redukcí). Pak z daného y, pro které hledáme x, dostáváme rovnice (nejméně $n_e - 1$ nezávislých rovnic) stupně 1 v hodnotách $x_1, ..., x_n$. Dle Gaussovy redukce najdeme $n_e - 1$ neznámých $x_1, ..., x_n$ z ostatních rovnic. Nyní si ukažme obecný případ.

Obecný případ
Mějme

$$b = a^{1+2^{m\theta}} \tag{3.20}$$

Složením obou stran této rovnice s $q : x \mapsto x^{2^{m\theta}-1}$, obdržíme $b^{2^{m\theta}-1} = a^{2^{2^{m\theta}}-1}$. Vynásobíme každou stranu ab

$$ab^{2^{m\theta}} = ba^{2^{2^{m\theta}}} \tag{3.21}$$

Nechť $(a_1, ..., a_{n_e})$ je reprezentace a v L_{n_e}, $(b_1, ..., b_{n_e})$ je reprezentace b v L_{n_e}. ($\forall a_i, b_i, 1 \leq \leq i \leq n_e$ jsou prvky v k). Rovnice (3.21) dává n_e rovnic (ne nutně nezávislých) stupně 1 na b_j hodnot a stupně 1 na a_j hodnot. (protože $b \mapsto b^{2^{m\theta}}$ je lineární, $a \mapsto a^{2^{2^{m\theta}}}$ je lineární) Navíc a je afinní v x, b je afinní v y. Tedy těchto n_e rovnic ((3.21) je báze) pak píšeme ve složkách $(x_1, ..., x_n)$, $(y_1, ..., y_n)$. x, y dává n_e rovnic tvaru

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\gamma_{ij}x_iy_j + \sum_{i=1}^{n}\alpha_ix_i + \sum_{i=1}^{n}\beta_iy_i + \delta_0 = 0 \tag{3.22}$$

Tyto rovnice platí $\forall x, y$, kde x je původní text y. Tedy vybráním hodnot pro x a spočítáním hodnot y z x a veřejného tvaru a pak nahrazením těchto hodnot $x_i, y_i, 1 \leq i \leq \leq n$ v (3.22) obdržíme rovnice stupně 1 v $(n+1)^2$ proměnných $GF(2^m) : \gamma_{ij}, \alpha_i, \beta_i, \delta_0$.

Tímto způsobem najdeme rychle všechny rovnice (3.22). To je první část našeho útoku. Možná najdeme některé rovnice (3.22), které nevychází z (3.21) (protože jsme našli všechny rovnice, které máme v obecném tvaru (3.22)), ale důležité je, že najdeme přinejmenším všechny rovnice (3.22), které vychází z (3.21).

Část 2 našeho útoku: Z daných y, pro které najdeme x, nám tyto rovnice dávají rovnice stupně 1 v neznámých $x_1, ..., x_n$. Gaussovou redukcí těchto rovnic najdeme λ neznámých $x_1, ..., x_n$ z ostatních, kde λ je počet nezávislých rovnic (3.22) v $x_1, ..., x_n$, když $y_1, .., y_n$ jsou nahrazeny hodnotami. Abychom vyčíslili řád tohoto útoku, musíme vyčíslit λ. To uděláme nyní.

Poznámka: Jestliže $m = 1, \theta_e = 1$, když najdeme všechny rovnice (3.22), najdeme všechny rovnice, které vychází z $b^2a = ba^4$ a všechny rovnice, které vychází z $ba = a^4$. Tyto rovnice nejsou formálně stejné (protože jestliže $b = 0$, první se zruší, druhá ne). Např. když $m = 1, \theta = 1, n_e = 5$, máme explicitně najít všechny rovnice (3.22). V tomto případě máme najít vektorový prostor řešení pro koeficienty $\gamma_{ij}, \alpha_i, \beta_i, \delta_0$ dimenze
přesně 10. V tomto případě $b^2a = ba^4$ a $ba = a^4$ dané 10 rovnicemi. Pak vybereme pro y danou hodnotu. Těchto 10 rovnic nám samozřejmě dá nejvýše 5 neznámých rovnic a jestliže pro toto y máme $b \neq 0$, pak nám dá přesně 4 nezávislé rovnice (protože máme přesně 2 řešení pro a).

Výpočet λ

Teorém 77. *Pro všechny účinné klíče a pro většinu šifrovaných textů y, počet λ nezávislých rovnic stupně 1 v $x_1, ..., x_n$ tak, že obdržíme z rovnic (3.22) pro toto dané y máme $\lambda \geq \sum_{e=1}^{d}(n_e - GCD(n_e, \theta_e)) \geq \frac{2n}{3}$. Navíc nám to ukazuje, že pro mnoho tajných klíčů a pro většinu šifrovaných textů máme $\lambda \geq n - d$.*

Lemma 78. *Nechť L je konečné pole s q prvky. Nechť p je clé číslo a nechť y je prvek L. Pak rovnice $x^p = y$ má nejvýše $GCD(p, q-1)$ řešení x.*

Důkaz. Jestliže $y = 0$, pak $x = 0$ je jediné řešení a Lemma 78 platí. Předpokládejme, že $y \neq 0$, pak $x = 0$ není řešení. Předpokládejme také, že $x \neq 0$, takže $x^{q-1} = 1$. Nechť $\mu = GCD(p, q-1)$. Z Bezoutova teorému víme, že jsou 2 celá čísla α, β taková, že $\alpha p - \beta(q-1) = \mu$. Tedy $x^p = y \Rightarrow x^{\alpha p} = y^\alpha \Rightarrow x^\mu (x^{q-1})^\beta = y^\alpha \Rightarrow x^\mu = y^\alpha$. Tedy v poli každá rovnice stupně k má nejvýše k řešení. Existuje nejvýše μ řešení $x^\mu = y^\alpha$ a je nejvýše μ řešení $x^p = y$, jak jsme tvrdili. □

Lemma 79. *Pro všechna celá čísla m, α, β máme $GCD(2^{m\alpha}-1, 2^{m\beta}-1) = 2^{mGCD(\alpha,\beta)}-1$.*

Důkaz.

- Jasně $GCD(2^{m\alpha}-1, 2^{m\beta}-1) \geq 2^{mGCD(\alpha,\beta)}-1$, protože $2^{m\alpha}-1$ a $2^{m\beta}-1$ můžeme vzít jako $2^{mGCD(\alpha,\beta)}-1$ faktor (použijeme formuli $x^k - 1 = (x-1)(x^{k-1} + x^{k-2} + \cdots + x + 1)$.
- Jasně také můžeme předpokládat, že $\alpha > \beta$. Jelikož Lemma 79 je symetrická, máme $\alpha = \beta$. Lemma 79 je zřejmá.
- Nyní jestliže x, y jsou 2 celá čísla a jestliže μ je celé číslo takové, že $x - y2^\mu > 0$, máme $GCD(x, y) = GCD(y, x - y2^\mu)$.

Tedy s $x = 2^{m\alpha} - 1, y = 2^{m\beta} - 1, 2^\mu = 2^{m(\alpha-\beta)}$ máme

$$GCD(2^{m\alpha} - 1, 2^{m\beta} - 1) = GCD(2^{m\beta} - 1, 2^{m(\alpha-\beta)} - 1) \tag{3.23}$$

Iterací této techniky $GCD(\alpha, \beta)$ objevíme způsob podobný Euklidově algoritmu. Takže obdržíme $GCD(2^{m\alpha} - 1, 2^{m\beta} - 1) \leq 2^{mGCD(\alpha,\beta)} - 1$ □

Lemma 80. *V L_{n_e} (pole s 2^{mn_e} prvky) rovnice (3.21), jak máme napsáno dříve, má nejvýše $2^{mGCD(\theta,n_e)}$ řešení v a, pro každé dané $b \neq 0$.*

Důkaz. Jestliže $b \neq 0$, pak tato rovnice (3.21) má 2 skupiny řešení pro a: 1)$a = 0$, 2)a tak, že

$$(a^{2^{m\theta}} - 1)^{2^{m\theta}+1} = b^{2^{m\theta}-1} \tag{3.24}$$

Víme, že funkce $g : z \mapsto z^{2^{m\theta}+1}$ je bijekce v L_{n_e} (protože dle konstrukce MI algoritmu, θ, n_e jsou vybrány tak, aby byla splněna daná vlastnost). Pak a je řešení rovnice (3.24)$\Leftrightarrow$

$$a^{2^{m\theta}} - 1 = g^{-1}(b^{2^{m\theta}} - 1) \tag{3.25}$$

Nyní z Lemmatu 78 víme, že tato rovnice (3.25) má pro dané b nejvíce $GCD(2^{m\theta}-1, 2^{mn_e}-1)$ řešení a. Takže z Lemmatu 79 obdržíme, že rovnice (3.24) má nejvíce $2^{mGCD(\theta,n_e)} - 1$ řešení v a. Takže přidáním řešení $a = 0$ obdržíme, že když $b \neq 0$, rovnice (3.21) má nejvíce $2^{mGCD(\theta,n_e)}$ řešení v a, jak bylo odvozeno. □

Důsledek 81. *Pro dané $b \neq 0$, jestliže napíšeme rovnici (3.21) v bázi prvků (s reprezentací L_{n_e} jako rozšíření $GF(2^m)$ stupně n_e), pak obdržíme nejméně $n_e - GCD(\theta, n_e)$ nezávislých rovnic stupně 1 v prvcích a.*

Důkaz. Víme, že obdržené rovnice jsou stupně 1 v prvcích a. Navíc tyto rovnice mají nejméně jedno řešení $a = 0$, takže nedochází ke sporu. Jestliže λ_e je počet nezávislých rovnic, máme přesně $2^{m(n_e-\lambda_e)}$ řešení. Avšak z Lemmatu 80 víme, že máme nejvýše $2^{mGCD(\theta,n_e)}$ řešení, takže $\lambda_e \geq n_e - GCD(\theta, n_e)$, jak jsme tvrdili. □

Důkaz. Teorému 77: Nechť y je šifrovaný text takový,že pro toto y máme: $\forall e, 1 \leq e \leq$ $\leq d, b_e \neq 0$. (Takže $a_e \neq 0$, protože $b_e = a_e^{1+2^{m\theta_e}}$.) □

Poznámka: Pro dané e je pravděpodobnost, že $b_e = 0$ rovna $\frac{1}{2^{mn_e}}$. Takže jestliže mn_e je velmi malé, tato pravděpodobnost není zanedbatelná. Avšak jesliže mn_e je velmi malé (např. $m = 1, n_e = 3$), pak $a_e = b_e^{\not{h}_e}$ s velmi malým $\not{h}_e$, takže s velmi malým $HW(\not{h}_e)$. To nám dává velmi slabé klíče. Můžeme předpokládat, že mn_e není tak malé, takže většina šifrovaných textů y bude taková, že $\forall e, 1 \leq e \leq d, b_e \neq 0$. (Pro velmi velké d nemůže být, ale jestliže n je přijatelné velikosti, pak d nemůže být příliš velké a pro většinu šifrovaných textů $y, \forall e, 1 \leq e \leq d, b_e \neq 0$ jak jsme tvrdili.) Z důsledku Lemmatu 80 víme, že obdržíme nejméně $\sum_{e=1}^{d}(n_e - GCD(\theta_e, n_e))$ nezávislých rovnic stupně 1 v prvcích x (pro dané y takové, že $\forall e, b_e \neq 0$).

Lemma 82. *$\forall e, 1 \leq e \leq d$, nechť $\delta_e = GCD(\theta_e, n_e)$ a nechť $k_e = \frac{n_e}{\delta_e}$ (k_e je celé číslo, protože δ_e dělí n_e). Pak k_e je vždy liché a $k_e \geq 3$.*

Dokázali jsme, že $\lambda \geq \sum_{e=1}^{d}(n_e - GCD(\theta_e, n_e))$. Takže z Lemmatu 82 plyne $\lambda \geq \sum_{e=1}^{d}(n_e - \frac{n_e}{3}) = \frac{2}{3}\sum_{e=1}^{d} n_e = \frac{2n}{3}$. Navíc pro mnoho tajných klíčů $GCD(\theta_e, n_e) = 1$ a jestliže toto nastane, máme $\sum_{e=1}^{d}(n_e - 1) = n - d$, takže $\lambda \geq n - d$.

Vylepšená Gaussova eliminace

Náš útok, jak jsme ho popsali, přechází na 2 části:

1. Najdeme všechny rovnice (3.22) a to je uděláno jednou a pro všechny.
2. Pro určitý šifrovaný text y se pokusíme najít x pomocí rovnic (3.22). Takže to musíme udělat pro každé x z různých y.

Druhý obecný útok

V odstavci 3.3.1 byl náš útok založen na myšlence, že jestliže $b = a^{1+2^{m\theta}}$, pak $ab^{2^{m\theta}} = ba^{2^{2^{m\theta}}}$. V této rovnici jsou a, b na obou stranách. Nyní se budeme snažit najít nějaké rovnice obecného tvaru $a^2 b^u = b^v$, kde $HW(u), HW(v)$ jsou malé. Pro tento záměr nezačneme z $b = a^{1+2^{m\theta}}$, ale začneme z $a = b^{\not{h}_e}$. Takže musíme spočítat hodnotu $\not{h}_e$.

Teorém 83. *Nechť $\delta = mGCD(\theta_e, n_e)$ a nechť α, k jsou celá čísla taková, že $\alpha\delta = m\theta_e$ a $k\delta = mn_e$. Nechť $\not{h}_e$ je, jako obvykle, multiplikativní inverze $1 + 2^{m\theta_e}$ modulo $2^{mn_e} - 1$. Pak*

1. *k je liché a $k \geq 3$*
2. *$\not{h}_e = 2^{k\delta-1} + \sum_{i=1}^{k-1}(-1)^i 2^{\alpha\delta_i - 1}$.*

Poznámka: Nejobtížnější klíče v odstavci 3.3.1 jsou klíče s malým k. Tyto klíče jsou nyní nejlehčí. Např. když $k = 3$, obdržíme n_e rovnic v obecném tvaru

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\gamma_{ij}x_i^2 y_j + \sum_{i=1}^{n}\sum_{j=i+1}^{n}\eta_{ij}y_i y_j + \sum_{i=1}^{n}\alpha_i x_i + \sum_{i=1}^{n}\beta_i y_i + \delta_0 = 0 \tag{3.26}$$

Náš útok má stále dvě části:

1. Najít všechny rovnice (3.22) z odstavce 3.3.1 a také všechny rovnice (3.26).
2. Pro dané y položme mocninu 2^{m-1} nalezených rovnic (3.26). Jelikož v k máme $(\alpha + \beta)^{2^{m-1}} = \alpha^{2^{m-1}} + \beta^{2^{m-1}}$ a tedy $x_i^{2^m} = x_i$, rovnice (3.26) dává podobné rovnice stupně 1 v x_i. Proto použijeme obě rovnice (3.22) a (3.26).

4. Implementace algoritmů

Při implementaci v prostředí Wolfram Mathematica bylo čerpáno z [7].

4.1. Buchbergerův algoritmus

Tento algoritmus se používá pro výpočet Gröbnerových bází a je v prostředí Wolfram Mathematica již implementován. Ukážeme si tedy, jak daný algoritmus spustit.

Při počítání nad reálnými čísly postupujeme následovně. Zadáme množinu polynomů

```
Poly={x^2 y - y, 2 x y + y, -5 + x z}
```

Zavoláme funkci "GroebnerBasis" a vypíšeme proměnné obsažené v polynomech. Pro náš příklad

```
GroebnerBasis[Poly,{x,y,z}]
```

Výstupem bude opět množina polynomů, a to redukovaná Gröbnerova báze. Pokud nezadáme žádné další parametry, je výsledná redukovaná Gröbnerova báze spočtena pro lexikografické uspořádání. Pokud bychom požadovali jiné uspořádání, např. stupňované inverzní lexikografické uspořádání, musíme změnit volání následovně

```
GroebnerBasis[Poly,{x,y,z},MonomialOrder->DegreeReverseLexicographic]
```

Při výpočtu Buchbergerova algoritmu nad konečnými poli musíme postupovat následovně. Nejdříve načteme balík s konečnými poli

```
<< FiniteFields`FiniteFields`
```

Dále musíme definovat námi vybrané pole včetně ireducibilního polynomu. Ten zapíšeme pomocí koeficientů. Např pro pole $\mathbb{F}_8$ a pro ireducibilní polynom [1,0,1,1] zadáme

```
SetFieldFormat[GF[2,{1,0,1,1}],FormatType->FunctionOfCode[k]]
```

Nyní již můžeme zadat množinu polynomů jen s tím rozdílem, že místo každého koeficientu i píšeme $k[i]$. Přepíšeme tedy náš příklad

```
Poly = {x^2 y - y, k[2] x y + y, k[-5] + x z}
```

Nyní již zavoláme funkci "GroebnerBasis" jako v předchozím

```
GroebnerBasis[Poly,{x,y,z}]
```

Zdá se tedy, že pro Gröbnerovy báze je v programu Wolfram Mathematica vše hotovo. Nastává zde ale problém. Tento algoritmus je dosti pomalý. Zabývali jsme se tedy implementací jiného algoritmu a to F4 algoritmu do prostředí Wolfram Mathematica.

4.2. F4 algoritmus

Tento algoritmus se používá pro výpočet Gröbnerových bází stejně jako Buchbergerův algoritmus. Oproti Buchbergerově algoritmu je ale výstupem z F4 algoritmu neredukovaná Gröbnerova báze. Tím ale nevzniká žádný problém, protože vytvořit redukovanou Gröbnerovu bázi je již snadný úkol. Výhodou tohoto algoritmu je především jeho rychlost. Ukažme tedy, jak jsme implementovali daný algoritmus do prostředí Wolfram Mathematica a jeho volání.

4.2.1. Implementace algoritmu

Pro výpočet Gröbnerových bází pomocí F4 algoritmu jsme vytvořili funkční balík v prostředí Wolfram Mathematica. K použití toho balíku musíme vytvořit složku s názvem "F4algorithm". Do této složky zkopírujeme balík "F4algorithm.m".

F4 algoritmus obsahuje dva podalgoritmy, a to redukci a symbolické přeprocesování. Pro každý podalgoritmus sestavíme vlastní funkci. Začněme tedy těmito podalgoritmy. Vstupem je zde množina zadaných polynomů a dále množina dvojic. Výstupem bude množina polynomů. Popíšeme si, co se v daných podalgoritmech počítá

- **Symbolické přeprocesování** V tomto podalgoritmu nejdříve roznásobíme všechny prvky v L, poté musíme vypsat všechny hlavní monomy dle daného monomického uspořádání. Dále musíme vypsat také všechny monomy a uděláme rozdíl těchto dvou množin. V dalším kroku spočítáme všechny hlavní monomy ze zadaných polynomů. Prozkoumáme top ireducibilitu. Výsledné polynomy pak přidáme k polynomům, které vznikly roznásobením L.

- **Redukce** Pro danou množinu polynomů sestavíme matici koeficientů. Dále tuto matici převedeme na řádkově schodovitý tvar a vypíšeme nově vzniklé polynomy. Pokud je některý hlavní monom nově vzniklých polynomů již obsažen v některém z hlavním monomů polynomů před redukcí, pak ho vynecháme, jinak ho do množiny polynomů přidáme.

V hlavním algoritmu je vstupem pouze množina polynomů. Výstupem pak množina polynomů a to neredukovaná Gröbnerova báze. V tomto algoritmu nejdříve musíme vypsat hlavní monomy daných polynomů dle daného monomického uspořádání. Poté spočítáme nejmenší společné násobky těchto monomů a sestavíme pětice dle definice. Naplníme zbývající proměnné a zavoláme funkci Redukce. Poté opět musíme určit hlavní monomy množiny polynomů a porovnat s hlavními monomy zadaných polynomů. Pokud se nějaký hlavní monom vyskytuje už mezi hlavními monomy ze zadaných polynomů, daný polynom nepřidáváme, jinak ho přidáme do výsledné množiny. Spočítáme také nové pětice. Vše opakuje do té doby, než je množina pětic nulová.

4.2.2. Použití algoritmu

F4 algoritmus jsme implementovali pouze pro počítání nad reálnými čísly a pro lexikografické uspořádání. Nejdříve si musíme nahrát balík F4algorithm.m takto

```
<< F4algorithm`F4algorithm`
```

Poté zadáme množinu polynomů

```
Poly={x^2 y - y, 2 x y + y, -5 + x z}
```

Pro ověření správnosti podalgoritmů zadáme ještě množinu dvojic. Pro náš příklad

```
L = {2 (-y + x^2 y), x (y + 2 x y)}
```

Nyní zavoláme jednotlivé podalgoritmy. Nejdříve spustíme Symbolické přeprocesování

```
SymbolicPreprocessing[L, Poly]
```

Výstupem bude množina polynomů

```
{y + 2 x y, -2 y + 2 x^2 y, x y + 2 x^2 y}
```

Dále spustíme funkci Redukce

```
F4Reduction[L, Poly]
```

Výstupem bude množina polynomů. V našem případě

```
{y}
```

Zavoláme funkci "F4alg". Pro náš příklad

```
F4alg[Poly]
```

Výstupem je opět množina polynomů a to neredukovaná Gröbnerova báze.

```
{y, y + 2 x y, -y + x^2 y}
```

Z této množiny vytvoříme redukovanou Gröbnerovu bázi snadno, pouze porovnáním hlavních monomů u daných polynomů. Vidíme, že hlavní monom druhého polynomu je x násobkem hlavního monomu prvního polynomu a hlavní monom třetího polynomu je x^2 násobkem hlavního monomu prvního polynomu. Tedy výsledná Gröbnerova báze obsahuje pouze jeden polynom a to y.

4.3. Čuang-c'ův algoritmus

Tento algoritmus převádí množinu polynomických zobrazení o větším počtu proměnných na jednu rovnici o jedné proměnné. Toho docílíme snadno pomocí Teorému 77. My jsme se však zabývali, zda se výsledek z Teorému 77 bude shodovat s výsledkem spočítaným pomocí Lagrangeova interpolačního polynomu. Zabývali jsme se tedy implementací Lagrangeovy interpolace. Pro počítání s rozšířeným polem jsem využila program Python, v němž je vše jednoduše interpretovatelné.

Nejdříve si musíme sestavit tabulky pro sčítání a násobení nad daným rozšířeným polem, v našem případě pro rozšířené pole $GF(16)$. Dále musíme sestavit tabulku hodnot pro množinu polynomických zobrazení. Nyní již přejděme k samotnému programu.

4.3.1. Popis jednotlivých funkcí

Celý program se skládá z 8 funkcí, které si v této části detailněji popíšeme.

sum_int
Tato funkce slouží k sečtení dvou čísel z dané aditivní tabulky. Např. pro součet čísel $2, 3$ bude vstup tvaru

```
sum_int(2,3)
```

Výstupem je tedy součet těchto čísel, tedy opět pouze číslo, v našem případě

```
1
```

Poznamenejme, že jsou zde zahrnuta všechna pravidla pro součet čísel nad daným rozšířeným konečným polem, a to $a = -a, a + a = 0, a + 0 = a, a + b = b + a$.

multiply_int
Tato funkce slouží k vynásobení dvou čísel z dané multiplikativní tabulky. Např. pro součin čísel $4, 8$ bude vstup tvaru

```
multiply_int(4,8)
```

Výstupem je tedy součin těchto čísel, tedy opět pouze číslo, v našem případě

```
9
```

I u této funkce jsou zahrnuta pravidla pro součin dvou čísel nad daným polem, a to $a = -a, a * 0 = 0, a * 1 = a, a * b = b * a$.

divide_int
Tato funkce slouží k vydělení dvou čísel z dané multiplikativní tabulky. Tato funkce projíždí danou multiplikativní tabulku a hledá, které číslo musí vynásobit s druhým ze zadaných čísel, aby dostala první číslo. Např. pro podíl čísel $4, 8$ bude vstup tvaru

```
divide_int(4,8)
```

Výstupem je tedy podíl těchto čísel, tedy opět pouze číslo, v našem případě

```
3
```

Zjistili jsme tedy, že podíl dvou čísel je roven 3, tedy pokud vynásobíme $3*8$, dostaneme číslo 4.

_multiply_poly
Tato funkce slouží k vynásobení dvou polynomů. Vstupem jsou tedy dva polynomy a výstupem bude jejich součin. Tato funkce je dále využita v součinu většího počtu polynomů. Ukážeme si ukázkový vstup i výstup. Vezměme si například polynomy $1+2x, 3+2x$. Polynomy zadáme pomocí koeficientů vzestupně dle mocniny u x, tedy ukázkový vstup je tvaru

```
_multiply_poly([1,2],[3,2])
```

Po spuštění dostaneme výstup tvaru

```
[3, 3, 3]
```

což zastupuje polynom $3 + 3x + 3x^2$.

multiply_poly
Tato funkce slouží k vynásobení libovolného počtu polynomů. Polynomy opět zapíšeme pomocí koeficientů u jednotlivých mocnin vzestupně. Chceme-li tedy vynásobit polynomy $1 + 2x, 3 + 2x, 3 + 2x + 2X^2$, musíme zadat následující

```
multiply_poly([1,2],[3,2],[3,2,2])
```

Výsledkem pak bude součin těchto polynomů, tedy

```
[2, 3, 2, 0, 1]
```

což zastupuje polynom $2 + 3x + 2x^2 + x^4$.

sum_poly
Tato funkce slouží k sečtení libovolného počtu polynomů. Chceme-li sečíst např. polynomy $1 + 2x + 2x^2, 3 + 2x + 4x^2, 4 + 5x^2 + x^3$, zapíšeme

```
sum_poly([1,2,2],[3,2,4],[4,0,5,1])
```

Výstupem pak bude součet daných polynomů

```
[6, 0, 3, 1]
```

což zastupuje polynom $6 + 3x^2 + x^3$.

print_poly
Tato funkce slouží k úpravě výstupu. Je- li výstupem polynom, pak jednotlivým koeficientům přiřadí příslušnou mocninu x. Nulové členy se nevypisují.

```
print_poly([3,0,2,1])
```

Výstup bude tedy tvaru

```
'3x^0 + 2x^2 + x^3'
```

calculate_lagrange
Toto je hlavní funkce celého programu. Musíme zde zadat funkční hodnoty ve všech bodech. V tomto algoritmu jsou využity všechny předchozí funkce pro výpočet jednotlivých l_s, které vynásobí danými funkčními hodnotami, což označíme L_s. Poté se již jednotlivé L_s sečtou a vyjde Lagrangeův interpolační polynom. Ukážeme si ukázkový výstup pro určité funkční hodnoty. Tyto hodnoty zapíšeme následovně

```
mapping = [9, 11, 11, 11, 13, 4, 10, 1, 13, 1, 4, 10, 13, 10, 1, 4],
```

kde první hodnota odpovídá bodu $x = 0$ a poslední odpovídá bodu $x = 16$. Poté již spustíme danou funkci

```
calculate_lagrange()
```

a dostaneme výstup tvaru

```
       [1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1]
l0  = ------------------------------------------------
      1
      [0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]
l1  = ------------------------------------------------
      1
      [0, 3, 2, 1, 3, 2, 1, 3, 2, 1, 3, 2, 1, 3, 2, 1]
l2  = ------------------------------------------------
      1
      [0, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1, 2, 3, 1]
l3  = ------------------------------------------------
      1
      [0, 10, 13, 9, 8, 2, 15, 6, 14, 12, 3, 5, 11, 7, 4, 1]
l4  = ------------------------------------------------------
      1
      [0, 8, 14, 11, 10, 2, 12, 7, 13, 15, 3, 4, 9, 6, 5, 1]
l5  = ------------------------------------------------------
      1
      [0, 14, 10, 12, 13, 3, 9, 5, 8, 11, 2, 7, 15, 4, 6, 1]
l6  = ------------------------------------------------------
      1
      [0, 13, 8, 15, 14, 3, 11, 4, 10, 9, 2, 6, 12, 5, 7, 1]
l7  = ------------------------------------------------------
      1
      [0, 5, 6, 9, 4, 3, 15, 13, 7, 12, 2, 10, 11, 14, 8, 1]
l8  = ------------------------------------------------------
      1
      [0, 11, 12, 15, 9, 1, 11, 12, 15, 9, 1, 11, 12, 15, 9, 1]
l9  = ----------------------------------------------------------
      1
      [0, 4, 7, 11, 5, 3, 12, 14, 6, 15, 2, 8, 9, 13, 10, 1]
l10 = ------------------------------------------------------
      1
      [0, 9, 15, 12, 11, 1, 9, 15, 12, 11, 1, 9, 15, 12, 11, 1]
l11 = ----------------------------------------------------------
      1
      [0, 15, 11, 9, 12, 1, 15, 11, 9, 12, 1, 15, 11, 9, 12, 1]
l12 = ----------------------------------------------------------
      1
```

```
      [0, 7, 5, 12, 6, 2, 9, 10, 4, 11, 3, 14, 15, 8, 13, 1]
l13 = ------------------------------------------------------
      1
      [0, 6, 4, 15, 7, 2, 11, 8, 5, 9, 3, 13, 12, 10, 14, 1]
l14 = ------------------------------------------------------
      1
      [0, 12, 9, 11, 15, 1, 12, 9, 11, 15, 1, 12, 9, 11, 15, 1]
l15 = ----------------------------------------------------------
      1
Ls = [[9, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 9],
      [0, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11],
      [0, 6, 13, 11, 6, 13, 11, 6, 13, 11, 6, 13, 11, 6, 13, 11],
      [0, 13, 6, 11, 13, 6, 11, 13, 6, 11, 13, 6, 11, 13, 6, 11],
      [0, 9, 8, 2, 15, 6, 14, 12, 3, 5, 11, 7, 4, 1, 10, 13],
      [0, 9, 6, 5, 1, 8, 14, 11, 10, 2, 12, 7, 13, 15, 3, 4],
      [0, 12, 13, 3, 9, 5, 8, 11, 2, 7, 15, 4, 6, 1, 14, 10],
      [0, 13, 8, 15, 14, 3, 11, 4, 10, 9, 2, 6, 12, 5, 7, 1],
      [0, 7, 12, 2, 10, 11, 14, 8, 1, 5, 6, 9, 4, 3, 15, 13],
      [0, 11, 12, 15, 9, 1, 11, 12, 15, 9, 1, 11, 12, 15, 9, 1],
      [0, 7, 11, 5, 3, 12, 14, 6, 15, 2, 8, 9, 13, 10, 1, 4],
      [0, 8, 6, 3, 7, 10, 8, 6, 3, 7, 10, 8, 6, 3, 7, 10],
      [0, 14, 4, 2, 5, 13, 14, 4, 2, 5, 13, 14, 4, 2, 5, 13],
      [0, 4, 11, 3, 14, 15, 8, 13, 1, 7, 5, 12, 6, 2, 9, 10],
      [0, 6, 4, 15, 7, 2, 11, 8, 5, 9, 3, 13, 12, 10, 14, 1],
      [0, 14, 13, 5, 2, 4, 14, 13, 5, 2, 4, 14, 13, 5, 2, 4]]
L = 9x^0 + 8x^6 + 2x^9 + 8x^12
```

Výsledný Lagrangeův polynom je tedy tvaru $8x^{12} + 2x^9 + 8x^6 + 9$.

5. Závěr

Cílem této diplomové práce bylo sestavit přehled komutativní algebry se zaměřením na Gröbnerovy báze. Dalším cílem bylo popsat multivariační kryptosystémy a ataky na ně. Posledním cílem byla implementace daných algoritmů.

Diplomová práce je členěna do tří kapitol. V první kapitole je sepsán přehled komutativní algebry zaměřený na Gröbnerovy báze. Je zde prostudována také problematika ideálů, monomického uspořádání a teorie eliminace. Ve druhé kapitole je sestaven přehled multivariačních kryptosystémů. Dále jsou zde studovány tři algoritmy. Buchbergerův algoritmus a F4 algoritmus využívající Gröbnerovy báze a posledním algoritmem je Čuang-c'ův algoritmus, který využívá rozšířená konečná pole a převod většího počtu multivariačních polynomických rovnic na jednu polynomickou rovnici o jedné proměnné. V poslední kapitole jsme se zabývali implementací daných algoritmů. Buchbergerův algoritmus je v prostředí Wolfram Mathematica již implementován, tudíž jsme se zabývali použitím daného algoritmu. F4 algoritmus jsme implementovali do programu Wolfram Mathematica. Byl vytvořen funkční balík řešící F4 algoritmus nad reálnými čísly pro lexikografické uspořádání. U posledního algoritmu byl vytvořen program v jazyce Python počítající Lagrangeův interpolační polynom nad rozšířeným konečným polem.

K diplomové práci jsou přiloženy příklady, v kterých jsou procvičovány ideály, Gröbnerovy báze, S-polynomy a implicitizace. Dále jsou přiloženy kódy a to pro F4 algoritmu a pro výpočet Lagrangeova interpolačního polynomu.

Literatura

[1] Cox, D., Little, J., O'Shea, D. *Ideals, Varieties, and Algorithms*. Springer Science+Business Media, LLC, Third edition, 2007.

[2] Faugère, J. Ch. *A new efficient algorithm for computing Gröbner bases (F4)*. LIP6/CNRS Université Paris VI, 1999.

[3] Patarin, J. *Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88*. CP8 TRANSAC, 1988.

[4] Ding, J., Gower, E. J., Schmidt, S. D. *Zhuang-Zi: A New Algorithm for Solving Multivariate Polynomial Equations over a Finite Field*. University of Cincinnati.

[5] Wang, Z.,Nie, X.,Zheng, S.,Yang, Y.,Zhang, Z. *A New Construction of Multivariate Public Key Encryption Scheme through Internally Perturbed Plus*. Beijing University of Posts and Telecommunications.

[6] Ding, J., Schmidt, D. *Multivariable Public–Key Cryptosystems*. University of Cincinnati.

[7] Wolfram Mathematica *Core Language*. Wolfram Research, Inc., 2008.

6. Příloha

6.1. Příklady

6.1.1. Ideály

Příklad 1:

Mějme ideál $\mathfrak{i} = (x^4, y^4, x^2y^2, x^2 + y^3)$. To znamená, že $\mathfrak{i}$ je generován množinou $G = \{x^4, y^4, x^2y^2, x^2 + y^3\}$. Každý prvek $\mathfrak{i}$ je tvaru $P(x,y)\,x^4 + Q(x,y)\,y^4 + R(x,y)\,x^2y^2 + S(x,y)\,(x^2 + y^3)$. Do $\mathfrak{i}$ patří:

$$\begin{gathered} x^4 \\ y^4 \\ x^2y^2 \\ x^2 + y^3 \quad / \cdot y \Rightarrow x^2y + y^4 \in \mathfrak{i} \Rightarrow x^2y \in \mathfrak{i} \end{gathered}$$

Tedy do $\mathfrak{i}$ patří:

$$\begin{gathered} x^4 \\ y^4 \\ x^2y \\ x^2 + y^3 \end{gathered}$$

Jelikož x^4 i x^2y jsou dělitelné x^2, což je hlavní monom u $x^2 + y^3$, můžeme je vynechat. Tzn. obecný prvek je tvaru $P(x,y)\,y^4 + Q(x,y)\,(x^2 + y^3)$. Pro obecný prvek tohoto tvaru zkoumáme jeho hlavní monom (LM). Zjistíme, že každý prvek můžeme vygenerovat vždy z $x^2 + y^3, y^4$. $\mathfrak{in}\,\mathfrak{i}$ je ideál generovaný LM libovolného polynomu z $\mathfrak{i}$. $\mathfrak{in}\,\mathfrak{i} = (x^2, y^4)$. Vezmeme-li LM z prvků g při lexikografickém uspořádání, máme $\{x^4, y^4, x^2y^2, x^2\}$, což není minimální množina generátorů. $LM(G) = \{x^2, y^4\}$. Tzn. G je Gröbnerova báze.

Příklad 2:

Mějme ideál $\mathfrak{i} = (x^4, y^4, x^2y^2, x^2 + xy^2)$. To znamená, že $\mathfrak{i}$ je generován množinou $G = \{x^4, y^4, x^2y^2, x^2 + xy^2\}$. Každý prvek $\mathfrak{i}$ je tvaru $P(x,y)\,x^4 + Q(x,y)\,y^4 + R(x,y)\,x^2y^2 + S(x,y)\,(x^2 + xy^2)$. Do $\mathfrak{i}$ patří:

$$\begin{gathered} x^4 \\ y^4 \\ x^2y^2 \\ x^2 + xy^2 \quad / \cdot x \Rightarrow x^3 + x^2y^2 \in \mathfrak{i} \Rightarrow x^3 \in \mathfrak{i} \Rightarrow x^4 \in \mathfrak{i}, x^2y^2 \in \mathfrak{i} \end{gathered}$$

Tedy do $\mathfrak{i}$ patří:

$$y^4$$
$$x^2 + xy^2$$

Tzn. obecný prvek je tvaru $P(x,y)\,y^4 + Q(x,y)\,(x^2 + xy^2)$. Pro obecný prvek tohoto tvaru zkoumáme jeho hlavní monom (LM). Zjistíme, že každý prvek můžeme vygenerovat vždy z $y^4, x^2 + xy^2$. Z toho plyne tvar počátečního ideálu $\mathfrak{in}\,\mathfrak{i} = (x^2 + xy^2, y^4) = (x^2, y^4)$. Vezmeme-li LM z prvků g při lexikografickém uspořádání, máme $\{x^4, y^4, x^2y^2, x^2\}$, což není minimální množina generátorů. $LM(G) = \{x^2, y^4\}$. Tzn. G je Gröbnerova báze.

Příklad 3:
Mějme ideál $\mathfrak{i} = (x^4, y^4, x^2y^2, x^2 + x^2y)$. To znamená, že $\mathfrak{i}$ je generován množinou $G = \{x^4, y^4, x^2y^2, x^2 + x^2y\}$. Každý prvek $\mathfrak{i}$ je tvaru $P(x,y)\,x^4 + Q(x,y)\,y^4 + R(x,y)\,x^2y^2 + S(x,y)\,(x^2 + x^2y)$. Do $\mathfrak{i}$ patří:

$$x^4$$
$$y^4$$
$$x^2y^2$$
$$x^2 + x^2y \quad / \cdot y \Rightarrow x^2y + x^2y^2 \in \mathfrak{i} \Rightarrow x^2y \in \mathfrak{i} \Rightarrow x^2 \in \mathfrak{i}, x^2y^2 \in \mathfrak{i}$$

Tedy do $\mathfrak{i}$ patří:

$$x^2$$
$$y^4$$

Tzn. obecný prvek je tvaru $P(x,y)\,x^2 + Q(x,y)\,y^4$. Pro obecný prvek tohoto tvaru zkoumáme jeho hlavní monom (LM). Zjistíme, že každý prvek můžeme vygenerovat vždy z x^2, y^4. Z toho plyne tvar počátečního ideálu $\mathfrak{in}\,\mathfrak{i} = (x^2, y^4)$. Vezmeme-li LM z prvků g při lexikografickém uspořádání, máme $\{x^4, y^4, x^2y^2, x^2\}$, což není minimální množina generátorů. $LM(G) = \{x^2, y^4\}$. Tzn. G je Gröbnerova báze.

Příklad 4:
Mějme ideál $\mathfrak{i} = (x^4, y^4, x^2y^2, x^2 + x^3)$. To znamená, že $\mathfrak{i}$ je generován množinou $G = \{x^4, y^4, x^2y^2, x^2 + x^3\}$. Každý prvek $\mathfrak{i}$ je tvaru $P(x,y)\,x^4 + Q(x,y)\,y^4 + R(x,y)\,x^2y^2 + S(x,y)\,(x^2 + x^3)$. Do $\mathfrak{i}$ patří:

$$x^4$$
$$y^4$$
$$x^2y^2$$
$$x^2 + x^3 \quad / \cdot x \Rightarrow x^3 \in \mathfrak{i} \Rightarrow x^2 \in \mathfrak{i}$$

Tedy do $\mathfrak{i}$ patří:

$$x^2$$
$$y^4$$

Tzn. obecný prvek je tvaru $P(x,y)\,x^2 + Q(x,y)\,y^4$. Pro obecný prvek tohoto tvaru zkoumáme jeho hlavní monom (LM). Zjistíme, že každý prvek můžeme vygenerovat vždy z x^2, y^4. Z toho plyne tvar počátečního ideálu $\mathfrak{in}\,\mathfrak{i} = (x^2, y^4)$. Vezmeme-li LM z prvků g při lexikografickém uspořádání, máme $\{x^4, y^4, x^2y^2, x^3\}$, což není minimální množina generátorů. $LM(G) = \{x^3, y^4, x^2y^2\}$. Tzn. G není Gröbnerova báze.

Příklad 5:

Mějme ideál $\mathfrak{i} = (x^4, y^4, x^2y^2, xy + x^3)$. To znamená, že $\mathfrak{i}$ je generován množinou $G = \{x^4, y^4, x^2y^2, xy + x^3\}$. Každý prvek $\mathfrak{i}$ je tvaru $P(x,y)\,x^4 + Q(x,y)\,y^4 + R(x,y)\,x^2y^2 + S(x,y)\,(xy + x^3)$. Do $\mathfrak{i}$ patří:

$$\begin{aligned} x^4 & \\ y^4 & \\ x^2y^2 & \\ xy + x^3 \quad & / \cdot x \Rightarrow x^2y \in \mathfrak{i}, / \cdot y \Rightarrow xy^2 \in \mathfrak{i} \end{aligned}$$

Tedy do $\mathfrak{i}$ patří:

$$xy^2$$
$$y^4$$
$$x^2y$$
$$x^3 + xy$$

Tzn. obecný prvek je tvaru $P(x,y)\,xy^2 + Q(x,y)\,y^4 + R(x,y)\,x^2y + S(x,y)\,(x^3 + xy)$. Pro obecný prvek tohoto tvaru zkoumáme jeho hlavní monom (LM). Zjistíme, že každý prvek můžeme vygenerovat vždy z $xy^2, y^4, x^2y, x^3 + xy$. Z toho plyne tvar počátečního ideálu $\mathfrak{in}\,\mathfrak{i} = (xy^2, y^4, x^2y, x^3 + xy) = (x^3, x^2y, xy^2, y^4)$. Vezmeme-li LM z prvků g při lexikografickém uspořádání, máme $\{x^4, y^4, x^2y^2, x^3\}$, což není minimální množina generátorů. $LM(G) = \{x^3, x^2y^2, y^4\}$. Tzn. G není Gröbnerova báze.

Příklad 6:

Mějme ideál $\mathfrak{i} = (x^4, y^4, x^2y^2, xy + xy^2)$. To znamená, že $\mathfrak{i}$ je generován množinou $G = \{x^4, y^4, x^2y^2, xy + xy^2\}$. Každý prvek $\mathfrak{i}$ je tvaru $P(x,y)\,x^4 + Q(x,y)\,y^4 + R(x,y)\,x^2y^2 + S(x,y)\,(xy + xy^2)$. Do $\mathfrak{i}$ patří:

$$\begin{array}{c} x^4 \\ y^4 \\ x^2y^2 \\ xy + xy^2 \quad / \cdot x \Rightarrow x^2y \in \mathfrak{i}, / \cdot y, / \cdot y^2 \Rightarrow xy^3 \in \mathfrak{i} \Rightarrow xy^2 \in \mathfrak{i} \Rightarrow xy \in \mathfrak{i} \end{array}$$

Tedy do $\mathfrak{i}$ patří:

$$\begin{array}{c} x^4 \\ xy \\ y^4 \end{array}$$

Tzn. obecný prvek je tvaru $P(x,y)\,x^4 + Q(x,y)\,xy + R(x,y)\,y^4$. Pro obecný prvek tohoto tvaru zkoumáme jeho hlavní monom (LM). Zjistíme, že každý prvek můžeme vygenerovat vždy z x^4, xy, y^4. Z toho plyne tvar počátečního ideálu $\mathfrak{in}\ \mathfrak{i} = (x^4, xy, y^4)$. Vezmeme-li LM z prvků g při lexikografickém uspořádání, máme $\{x^4, y^4, x^2y^2, xy^2\}$, což není minimální množina generátorů. $LM(G) = \{x^4, xy^2, y^4\}$. Tzn. G není Gröbnerova báze.

Příklad 7:

Mějme ideál $\mathfrak{i} = (x^4, y^4, x^2y^2, xy + x^2y)$. To znamená, že $\mathfrak{i}$ je generován množinou $G = \{x^4, y^4, x^2y^2, xy + x^2y\}$. Každý prvek $\mathfrak{i}$ je tvaru $P(x,y)\,x^4 + Q(x,y)\,y^4 + R(x,y)\,x^2y^2 + S(x,y)\,(xy + x^2y)$. Do $\mathfrak{i}$ patří:

$$\begin{array}{c} x^4 \\ y^4 \\ x^2y^2 \\ xy + x^2y \quad / \cdot y \Rightarrow xy^2 \in \mathfrak{i}, / \cdot x \Rightarrow x^2y + x^3y \in \mathfrak{i} \Rightarrow x^2y \in \mathfrak{i} \Rightarrow xy \in \mathfrak{i} \end{array}$$

Tedy do $\mathfrak{i}$ patří:

$$\begin{array}{c} x^4 \\ xy \\ y^4 \end{array}$$

Tzn. obecný prvek je tvaru $P(x,y)\,x^4 + Q(x,y)\,xy + R(x,y)\,y^4$. Pro obecný prvek tohoto tvaru zkoumáme jeho hlavní monom (LM). Zjistíme, že každý prvek můžeme vygenerovat vždy z x^4, xy, y^4. Z toho plyne tvar počátečního ideálu $\mathfrak{in}\ \mathfrak{i} = (x^4, xy, y^4)$.

Vezmeme-li LM z prvků g při lexikografickém uspořádání, máme $\{x^4, y^4, x^2y^2, x^2y\}$, což není minimální množina generátorů. $LM(G) = \{x^4, x^2y, y^4\}$. Tzn. G není Gröbnerova báze.

Příklad 8:

Mějme ideál $\mathfrak{i} = (x^4, y^4, x^2y^2, xy + y^3)$. To znamená, že $\mathfrak{i}$ je generován množinou $G = \{x^4, y^4, x^2y^2, xy + y^3\}$. Každý prvek $\mathfrak{i}$ je tvaru $P(x,y)\,x^4 + Q(x,y)\,y^4 + R(x,y)\,x^2y^2 + S(x,y)\,(xy + y^3)$. Do $\mathfrak{i}$ patří:

$$\begin{gathered} x^4 \\ y^4 \\ x^2y^2 \\ xy + y^3 \quad / \cdot y \Rightarrow xy^2 \in \mathfrak{i} \end{gathered}$$

Tedy do $\mathfrak{i}$ patří:

$$\begin{gathered} x^4 \\ xy + y^3 \\ y^4 \end{gathered}$$

Tzn. obecný prvek je tvaru $P(x,y)\,x^4 + Q(x,y)\,(xy + y^3) + R(x,y)\,y^4$. Pro obecný prvek tohoto tvaru zkoumáme jeho hlavní monom (LM). Zjistíme, že každý prvek můžeme vygenerovat vždy z $x^4, xy + y^3, y^4$. Z toho plyne tvar počátečního ideálu $\mathfrak{in}\ \mathfrak{i} = (x^4, xy + y^3, y^4) = (x^4, xy, y^4)$. Vezmeme-li LM z prvků g při lexikografickém uspořádání, máme $\{x^4, y^4, x^2y^2, xy\}$, což není minimální množina generátorů. $LM(G) = \{x^4, xy, y^4\}$. Tzn. G je Gröbnerova báze.

Příklad 9:

Mějme ideál $\mathfrak{i} = (x^4, y^4, x^2y^2, x^3 + y^2)$. To znamená, že $\mathfrak{i}$ je generován množinou $G = \{x^4, y^4, x^2y^2, x^3 + y^2\}$. Každý prvek $\mathfrak{i}$ je tvaru $P(x,y)\,x^4 + Q(x,y)\,y^4 + R(x,y)\,x^2y^2 + S(x,y)\,(x^3 + y^2)$. Do $\mathfrak{i}$ patří:

$$\begin{gathered} x^4 \\ y^4 \\ x^2y^2 \\ x^3 + y^2 \quad / \cdot x \Rightarrow xy^2 \in \mathfrak{i} \end{gathered}$$

Tedy do $\mathfrak{i}$ patří:

$$\begin{gathered} xy^2 \\ x^3 + y^2 \\ y^4 \end{gathered}$$

Tzn. obecný prvek je tvaru $P(x,y)\,xy^2+Q(x,y)\,(x^3+y^2)+R(x,y)\,y^4$. Pro obecný prvek tohoto tvaru zkoumáme jeho hlavní monom (LM). Zjistíme, že každý prvek můžeme vygenerovat vždy z xy^2, x^3+y^2, y^4. Z toho plyne tvar počátečního ideálu
$\mathfrak{in}\,\mathfrak{i} = (xy^2, x^3+y^2, y^4) = (x^3, xy^2, y^4)$. Vezmeme-li LM z prvků g při lexikografickém uspořádání, máme $\{x^4, y^4, x^2y^2, x^3\}$, což není minimální množina generátorů. $LM(G) = \{x^3, x^2y^2, y^4\}$. Tzn. G není Gröbnerova báze.

Příklad 10:
Mějme ideál $\mathfrak{i} = (x^4, y^4, x^2y^2, xy^2+y^2)$. To znamená, že $\mathfrak{i}$ je generován množinou $G = \{x^4, y^4, x^2y^2, xy^2+y^2\}$. Každý prvek $\mathfrak{i}$ je tvaru $P(x,y)\,x^4+Q(x,y)\,y^4+R(x,y)\,x^2y^2+S(x,y)\,(xy^2+y^2)$. Do $\mathfrak{i}$ patří:

$$\begin{gathered} x^4 \\ y^4 \\ x^2y^2 \\ xy^2+y^2 \quad / \cdot x \Rightarrow xy^2 \in \mathfrak{i} \Rightarrow y^2 \in \mathfrak{i} \end{gathered}$$

Tedy do $\mathfrak{i}$ patří:

$$\begin{gathered} y^2 \\ x^4 \end{gathered}$$

Tzn. obecný prvek je tvaru $P(x,y)\,y^2 + Q(x,y)\,x^4$. Pro obecný prvek tohoto tvaru zkoumáme jeho hlavní monom (LM). Zjistíme, že každý prvek můžeme vygenerovat vždy z y^2, x^4. Z toho plyne tvar počátečního ideálu $\mathfrak{in}\,\mathfrak{i} = (y^2, x^4)$. Vezmeme-li LM z prvků g při lexikografickém uspořádání, máme $\{x^4, y^4, x^2y^2, xy^2\}$, což není minimální množina generátorů. $LM(G) = \{x^4, xy^2, y^4\}$. Tzn. G není Gröbnerova báze.

Příklad 11:
Mějme ideál $\mathfrak{i} = (x^4, y^4, x^2y^2, x^2y+y^2)$. To znamená, že $\mathfrak{i}$ je generován množinou $G = \{x^4, y^4, x^2y^2, x^2y+y^2\}$. Každý prvek $\mathfrak{i}$ je tvaru $P(x,y)\,x^4+Q(x,y)\,y^4+R(x,y)\,x^2y^2+S(x,y)\,(x^2y+y^2)$. Do $\mathfrak{i}$ patří:

$$\begin{gathered} x^4 \\ y^4 \\ x^2y^2 \\ x^2y+y^2 \quad / \cdot y \Rightarrow y^3 \in \mathfrak{i} \end{gathered}$$

Tedy do $\mathfrak{i}$ patří:

$$\begin{gathered} y^3 \\ x^4 \\ x^2y+y^2 \end{gathered}$$

Tzn. obecný prvek je tvaru $P(x,y)\,y^3+Q(x,y)\,x^4+R(x,y)\,(x^2y+y^2)$. Pro obecný prvek tohoto tvaru zkoumáme jeho hlavní monom (LM). Zjistíme, že každý prvek můžeme vygenerovat vždy z y^3, x^4, x^2y+y^2. Z toho plyne tvar počátečního ideálu $\mathfrak{in}\ \mathfrak{i} = (y^3, x^4, x^2y+y^2) = (x^4, x^2y, y^4)$. Vezmeme-li LM z prvků g při lexikografickém uspořádání, máme $\{x^4, y^4, x^2y^2, x^2y\}$, což není minimální množina generátorů. $LM(G) = \{x^4, x^2y, y^4\}$. Tzn. G není Gröbnerova báze.

Příklad 12:

Mějme ideál $\mathfrak{i} = (x^4, y^4, x^2y^2, y^3+y^2)$. To znamená, že $\mathfrak{i}$ je generován množinou $G = \{x^4, y^4, x^2y^2, y^3+y^2\}$. Každý prvek $\mathfrak{i}$ je tvaru $P(x,y)\,x^4 + Q(x,y)\,y^4 + R(x,y)\,x^2y^2 + S(x,y)\,(y^3+y^2)$. Do $\mathfrak{i}$ patří:

$$\begin{aligned} &x^4 \\ &y^4 \\ &x^2y^2 \\ &y^3+y^2 \quad / \cdot y \Rightarrow y^3 \in \mathfrak{i} \Rightarrow y^2 \in \mathfrak{i} \end{aligned}$$

Tedy do $\mathfrak{i}$ patří:

$$\begin{aligned} &y^2 \\ &x^4 \end{aligned}$$

Tzn. obecný prvek je tvaru $P(x,y)\,y^2 + Q(x,y)\,x^4$. Pro obecný prvek tohoto tvaru zkoumáme jeho hlavní monom (LM). Zjistíme, že každý prvek můžeme vygenerovat vždy z y^2, x^4. Z toho plyne tvar počátečního ideálu $\mathfrak{in}\ \mathfrak{i} = (y^2, x^4)$. Vezmeme-li LM z prvků g při lexikografickém uspořádání, máme $\{x^4, y^4, x^2y^2, y^3\}$, což není minimální množina generátorů. $LM(G) = \{x^4, x^2y^2, y^3\}$. Tzn. G není Gröbnerova báze.

6.1.2. Implicitizace

Implicitizace naivním způsobem, to jest určením stupně výsledného polynomu Q.

Příklad 1: $x = t + t^2, y = 2t^2$.

Uvažujme, že Q je polynom 2. stupně. Obecně tedy můžeme polynom Q zapsat ve tvaru: $Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0$. Za x, y dosadíme výrazy ze zadání a dostaneme $A(t+t^2)^2 + B(t+t^2)2t^2 + 4Ct^4 + D(t+t^2) + 2Et^2 + F = 0$.

Vyřešíme porovnáním koeficientů u jednotlivých mocnin:

$$\begin{aligned} t^4 : 2A + 2B + 4C &= 0 \\ t^3 : \quad 2A + 2B &= 0 \\ t^2 : \quad A + D + 2E &= 0 \\ t^1 : \quad D &= 0 \\ t^0 : \quad F &= 0 \end{aligned}$$

Vyřešením rovnic dostáváme: $A = -B, A = -2E, A = 4C, D = 0, F = 0$. Zvolíme $A = 4$. Pak výsledný polynom Q je tvaru $4x^2 - 4xy + y^2 - 2y = 0$.

Příklad 2: $x = t + t^3, y = t^2 + 1$.
Uvažujme, že Q je polynom 3. stupně. Obecně tedy můžeme polynom Q zapsat ve tvaru: $Ax^3 + Bx^2y + Cxy^2 + Dy^3 + Ex^2 + Fxy + Gy^2 + Hx + Iy + J = 0$. Za x, y dosadíme výrazy ze zadání a dostaneme $A(t+t^3)^3 + B(t+t^3)^2(t^2+1) + C(t+t^3)(t^2+1)^2 + D(t^2+1)^3 + E(t+t^3)^2 + F(t+t^3)(t^2+1) + G(t^2+1)^2 + H(t+t^3) + I(t^2+1) + J = 0$.
Vyřešíme porovnáním koeficientů u jednotlivých mocnin:

$$\begin{aligned}
t^9 : \quad & A = 0 \\
t^8 : \quad & B = 0 \\
t^7 : \quad & 3A + C = 0 \\
t^6 : \quad & 3B + D + E = 0 \\
t^5 : \quad & 3A + 3C + F = 0 \\
t^4 : \quad & 3B + 3D + 2E + G = 0 \\
t^3 : \quad & A + 3C + 2F + H = 0 \\
t^2 : \quad & B + 3D + E + 2G + I = 0 \\
t^1 : \quad & C + F + H = 0 \\
t^0 : \quad & D + G + I + J = 0
\end{aligned}$$

Vyřešením rovnic dostáváme: $A = 0, B = 0, C = 0, D = -E, F = 0, G = E, I = 0, F = 0, H = 0, J = 0$. Zvolíme $D = 1$. Pak výsledný polynom Q je tvaru $y^3 - x^2 - y^2 = 0$.

Příklad 3: $x = 1 + t^3, y = t^2 + t$.
Uvažujme, že Q je polynom 3. stupně. Obecně tedy můžeme polynom Q zapsat ve tvaru: $Ax^3 + Bx^2y + Cxy^2 + Dy^3 + Ex^2 + Fxy + Gy^2 + Hx + Iy + J = 0$. Za x, y dosadíme výrazy ze zadání a dostaneme $A(1+t^3)^3 + B(1+t^3)^2(t^2+t) + C(1+t^3)(t^2+t)^2 + D(t^2+t)^3 + E(1+t^3)^2 + F(1+t^3)(t^2+t) + G(t^2+t)^2 + H(1+t^3) + I(t^2+t) + J = 0$.
Vyřešíme porovnáním koeficientů u jednotlivých mocnin:

$$\begin{aligned}
t^9 : \quad & A = 0 \\
t^8 : \quad & B = 0 \\
t^7 : \quad & B + C = 0 \\
t^6 : \quad & 3A + 2C + D + E = 0 \\
t^5 : \quad & 2B + C + 3D + F = 0 \\
t^4 : \quad & 2B + C + 3D + F + G = 0 \\
t^3 : \quad & 3A + 2C + D + 2E + 2G + H = 0 \\
t^2 : \quad & B + C + F + G + I = 0 \\
t^1 : \quad & B + F + I = 0 \\
t^0 : \quad & A + E + H + J = 0
\end{aligned}$$

Vyřešením rovnic dostáváme: $A = 0, B = 0, C = 0, D = -E, 3D = -F, G = 0, E = -H, F = -I, J = 0$. Zvolíme $E = -1$. Pak výsledný polynom Q je tvaru $y^3 - x^2 - 3xy + x + 3y = 0$.

Příklad 4: $x = t^3 - 1, y = t^2 + t$.
Uvažujme, že Q je polynom 3. stupně. Obecně tedy můžeme polynom Q zapsat ve tvaru: $Ax^3 + Bx^2y + Cxy^2 + Dy^3 + Ex^2 + Fxy + Gy^2 + Hx + Iy + J = 0$. Za x, y dosadíme výrazy ze zadání a dostaneme $A(t^3-1)^3 + B(t^3-1)^2(t^2+t) + C(t^3-1)(t^2+t)^2 + D(t^2+t)^3 + E(t^3-1)^2 + F(t^3-1)(t^2+t) + G(t^2+t)^2 + H(t^3-1) + I(t^2+t) + J = 0$.
Vyřešíme porovnáním koeficientů u jednotlivých mocnin:

$$\begin{aligned}
t^9 &: & A &= 0\\
t^8 &: & B &= 0\\
t^7 &: & B + C &= 0\\
t^6 &: & -3A + 2C + D + E &= 0\\
t^5 &: & -2B + C + 3D + F &= 0\\
t^4 &: & -2B - C + 3D + F + G &= 0\\
t^3 &: & 3A - 2C + D - 2E + 2G + H &= 0\\
t^2 &: & B - C - F + G + I &= 0\\
t^1 &: & B - F + I &= 0\\
t^0 &: & -A + E - H + J &= 0
\end{aligned}$$

Vyřešením rovnic dostáváme: $A = 0, B = 0, C = 0, D = -E, 3D = -F, 3E = H, G = 0, F = I, J = 2E$. Zvolíme $E = -1$. Pak výsledný polynom Q je tvaru $y^3 - x^2 - 3xy - 3x - 3y - 2 = 0$.

6.1.3. S-polynomy

Příklad 1: $P = x^4, Q = y^4, R = x^2y^2, T = x^2 + y^3$. Spočítejte S-polynomy.

$$S(P,Q) = \frac{x^4y^4}{x^4}x^4 - \frac{x^4y^4}{y^4}y^4 = 0$$

$$S(P,R) = \frac{x^4y^2}{x^4}x^4 - \frac{x^4y^2}{x^2y^2}x^2y^2 = 0$$

$$S(P,T) = \frac{x^4}{x^4}x^4 - \frac{x^4}{x^2}(x^2 + y^3) = x^2y^3 = yR$$

$$S(Q,R) = \frac{x^2y^4}{y^4}y^4 - \frac{x^2y^4}{x^2y^2}x^2y^2 = 0$$

$$S(Q,T) = \frac{x^2y^4}{y^4}y^4 - \frac{x^2y^4}{x^2}(x^2 + y^3) = y^7 = y^3Q$$

$$S(R,T) = \frac{x^2y^2}{x^2y^2}x^2y^2 - \frac{x^2y^2}{x^2}(x^2 + y^3) = y^5 = yQ$$

Nedostali jsme žádný nový polynom $\Rightarrow G = \{x^2 + y^3, y^4\}$. Z tohoto příkladu vidíme, že jediné nenulové polynomy dostaneme pouze v kombinaci s polynomem T, čehož využijeme v následujících úlohách, kde již zbývající kombinace počítat nebudeme.

Příklad 4: $P = x^4, Q = y^4, R = x^2y^2, T = x^2 + x^3$. Spočítejte S-polynomy.

$$S(P,T) = \frac{x^4}{x^4}x^4 - \frac{x^4}{x^3}(x^3 + x^2) = x^3 = U$$

$$S(Q,T) = \frac{x^3y^4}{y^4}y^4 - \frac{x^3y^4}{x^3}(x^3 + x^2) = x^2y^4 = y^2R$$

$$S(R,T) = \frac{x^3y^2}{x^2y^2}x^2y^2 - \frac{x^3y^2}{x^3}(x^3 + x^2) = x^3y^2 = xR$$

$$S(U,T) = \frac{x^3}{x^3}x^3 - \frac{x^3}{x^3}(x^3 + x^2) = x^2 = V$$

Ještě bychom měli vypočítat $S(V,T)$, ale jak je již vidět, nic nového už nevyjde. V tomto případě je $\Rightarrow G = \{x^2, y^4\}$.

Příklad 5: $P = x^4, Q = y^4, R = x^2y^2, T = xy + x^3$. Spočítejte S-polynomy.

$$S(P,T) = \frac{x^4}{x^4}x^4 - \frac{x^4}{x^3}(x^3 + xy) = x^2y = U$$

$$S(Q,T) = \frac{x^3y^4}{y^4}y^4 - \frac{x^3y^4}{x^3}(x^3 + xy) = xy^5 = V = y^2W = y^3Z$$

$$S(R,T) = \frac{x^3y^2}{x^2y^2}x^2y^2 - \frac{x^3y^2}{x^3}(x^3 + xy) = xy^3 = W = yZ$$

$$S(U,T) = \frac{x^3y}{xy^3}xy^3 - \frac{x^3y}{x^3}(x^3 + xy) = xy^2 = Z$$

$$S(Z,T) = \frac{x^3y^2}{xy^3}xy^3 - \frac{x^3y^2}{x^3}(x^3 + xy) = xy^3 = W$$

V tomto případě je $\Rightarrow G = \{x^3 + xy, xy^2, x^2y, y^4\}$.

Příklad 6: $P = x^4, Q = y^4, R = x^2y^2, T = xy^2 + xy$. Spočítejte S-polynomy.

$$S(P,T) = \frac{x^4y^2}{x^4}x^4 - \frac{x^4y^2}{xy^2}(xy^2 + xy) = x^4y = yP$$

$$S(Q,T) = \frac{xy^4}{y^4}y^4 - \frac{xy^4}{xy^2}(xy^2 + xy) = xy^3 = U = yW = y^2Z$$

$$S(R,T) = \frac{x^2y^2}{x^2y^2}x^2y^2 - \frac{x^2y^2}{xy^2}(xy^2 + xy) = x^2y = V$$

$$S(U,T) = \frac{xy^3}{xy^3}xy^3 - \frac{xy^3}{xy^2}(xy^2 + xy) = xy^2 = W = yZ$$

$$S(V,T) = \frac{x^2y^2}{x^2y}x^2y - \frac{x^2y^2}{xy^2}(xy^2 + xy) = x^2y = V$$

$$S(W,T) = \frac{xy^2}{xy^2}xy^2 - \frac{xy^2}{xy^2}(xy^2 + xy) = xy = Z$$

V tomto případě je $\Rightarrow G = \{x^4, xy, y^4\}$.

Příklad 7: $P = x^4, Q = y^4, R = x^2y^2, T = x^2y + xy$. Spočítejte S-polynomy.

$$S(P,T) = \frac{x^4y}{x^4}x^4 - \frac{x^4y}{x^2y}(x^2y + xy) = x^3y = U$$

$$S(Q,T) = \frac{x^2y^4}{y^4}y^4 - \frac{x^2y^4}{x^2y}(x^2y + xy) = xy^4 = y^2V$$

$$S(R,T) = \frac{x^2y^2}{x^2y^2}x^2y^2 - \frac{x^2y^2}{x^2y}(x^2y + xy) = xy^2 = V$$

$$S(U,T) = \frac{x^3y}{x^3y}x^3y - \frac{x^3y}{x^2y}(x^2y + xy) = x^2y = Z = xW$$

$$S(V,T) = \frac{x^2y^2}{xy^2}xy^2 - \frac{x^2y^2}{x^2y}(x^2y + xy) = xy^2 = V$$

$$S(Z,T) = \frac{x^2y}{x^2y}x^2y - \frac{x^2y}{x^2y}(x^2y + xy) = xy = W$$

V tomto případě je $\Rightarrow G = \{x^4, xy, y^4\}$.

Příklad 9: $P = x^4, Q = y^4, R = x^2y^2, T = x^3 + y^2$. Spočítejte S-polynomy.

$$S(P,T) = \frac{x^4y}{x^4}x^4 - \frac{x^4y}{x^3}(x^3 + y^2) = xy^2 = U$$

$$S(Q,T) = \frac{x^3y^4}{y^4}y^4 - \frac{x^3y^4}{x^3}(x^3 + y^2) = y^6 = y^2Q$$

$$S(R,T) = \frac{x^3y^2}{x^2y^2}x^2y^2 - \frac{x^3y^2}{x^3}(x^3 + y^2) = y^4 = Q$$

$$S(U,T) = \frac{x^3y^2}{xy^2}xy^2 - \frac{x^3y^2}{x^3}(x^3 + y^2) = y^4 = Q$$

V tomto případě je $\Rightarrow G = \{x^3 + y^2, xy^2, y^4\}$.

Příklad 10: $P = x^4, Q = y^4, R = x^2y^2, T = xy^2 + y^2$. Spočítejte S-polynomy.

$$S(P,T) = \frac{x^4y^2}{x^4}x^4 - \frac{x^4y^2}{xy^2}(xy^2 + y^2) = x^3y^2 = xR$$

$$S(Q,T) = \frac{xy^4}{y^4}y^4 - \frac{xy^4}{xy^2}(xy^2 + y^2) = y^4 = Q$$

$$S(R,T) = \frac{x^2y^2}{x^2y^2}x^2y^2 - \frac{x^2y^2}{xy^2}(xy^2 + y^2) = xy^2 = U$$

$$S(U,T) = \frac{xy^2}{xy^2}xy^2 - \frac{xy^2}{xy^2}(xy^2 + y^2) = y^2$$

V tomto případě je $\Rightarrow G = \{x^4, y^2\}$.

Příklad 11: $P = x^4, Q = y^4, R = x^2y^2, T = x^2y + y^2$. Spočítejte S-polynomy.

$$S(P,T) = \frac{x^4y}{x^4}x^4 - \frac{x^4y}{x^2y}(x^2y + y^2) = x^2y^2 = R$$

$$S(Q,T) = \frac{x^2y^4}{y^4}y^4 - \frac{x^2y^4}{x^2y}(x^2y + y^2) = y^5 = yQ$$

$$S(R,T) = \frac{x^2y^2}{x^2y^2}x^2y^2 - \frac{x^2y^2}{x^2y}(x^2y + y^2) = y^3 = U$$

$$S(U,T) = \frac{x^2y^3}{y^3}y^3 - \frac{x^2y^3}{x^2y}(x^2y + y^2) = y^4 = Q$$

V tomto případě je $\Rightarrow G = \{x^4, x^2y + y^2, y^3\}$.

Příklad 12: $P = x^4, Q = y^4, R = x^2y^2, T = y^3 + y^2$. Spočítejte S-polynomy.

$$S(P,T) = \frac{x^4y^3}{x^4}x^4 - \frac{x^4y^3}{y^3}(y^3 + y^2) = x^4y^2 = y^2P$$

$$S(Q,T) = \frac{y^4}{y^4}y^4 - \frac{y^4}{y^3}(y^3 + y^2) = y^3 = U$$

$$S(R,T) = \frac{x^2y^3}{x^2y^2}x^2y^2 - \frac{x^2y^3}{y^3}(y^3 + y^2) = x^2y^2 = R$$

$$S(U,T) = \frac{y^3}{y^3}y^3 - \frac{y^3}{y^3}(y^3 + y^2) = y^2 = V$$

$$S(V,T) = \frac{y^3}{y^2}y^2 - \frac{y^3}{y^3}(y^3 + y^2) = y^2 = V$$

V tomto případě je $\Rightarrow G = \{x^4, y^2\}$.

6.2. Kódy

6.2.1. F4 algoritmus

```
BeginPackage["F4algorithm`"]

F4alg::usage=
"F4 nám dává ze zadané množiny polynomů Gröbnerovu bázi pomocí
F4 algoritmu."
F4Reduction::usage="počítá redukci"
SymbolicPreprocessing::usage="symbolické přeprocesování"
```

```
Begin["`Private`"]

Unprotect[SymbolicPreprocessing, F4Reduction, F4alg]

SymbolicPreprocessing[LL_List, GG_List] :=
  Module[{F, Done, TF, M, lt, MFFF, MNove, Fkonecne,
    A, TFF},(*Funkce SymbolicPreprocessing.
   Vstupem je množina polynomů GG a množina čtveřic LL.
   Výstupem je množina polynomů Fkonecne.*)

   F = Expand[LL];(*Roznásobí LL*)

   Done = DeleteDuplicates[
     Map[If[#[[0]] === Times , If[NumericQ[#[[1]]], #/#[[1]], #],
        If[NumericQ[#], 1, #]] &, MonomialList[F][[All, 1]]]];
   (*Vypíše hlavní monomy z F.
   Nejdříve sestavíme MonomialList z F a vybereme první člen.
   Pokud je tento člen typu Times(musí zde být tři rovnítka,
   jelikož porovnáváme neurčité a ne konkrétní čísla), zkoumáme,
   zda je první část tohoto člene číslo. Pokud ano,
   tak tento člen číslem podělíme,
   abychom se tak zbavili koeficientu. Pokud ne,
   tak vypíšeme celý člen. Pokud člen není typu Times, zkoumáme,
   zda je to číslo. Pokud ano, napíšeme 1, pokud ne,
   napíšeme celý člen.*)
   TFF = DeleteDuplicates[Map[(If[#[[0]] === Times , If[NumericQ[#[[1]]],
   #/#[[1]], #], If[NumericQ[#], 1, #]]) &,
   Union[Flatten[F /. Plus -> List]]]] /. List -> Plus;

   TF = MonomialList[TFF];

(*Vypíše množinu všech monomů obsažených v F. Opět musíme rozlišit,
   zda je člen typu Times. Pokud ano,
   tak pokud je to první část číslo,
   tak jím podělíme a odstraníme tak koeficient. Pokud ne,
   vypíšeme celý člen. Dále pokud člen není typu Times,
   ale je to číslo, vypíšeme 1, jinak vypíšeme daný člen.*)
   M = Complement[TF, Done];(*Doplněk TF,Done.
   Tímto získáváme množinu M={m_{i}}*)
   lt = DeleteDuplicates[
     Map[If[#[[0]] === Times , If[NumericQ[#[[1]]], #/#[[1]], #],
        If[NumericQ[#], 1, #]] &, MonomialList[GG][[All, 1]]]];
   (*Vypíšeme hlavní členy ze zadaných polynomů z GG,
   avšak zase musíme odstranit koeficienty.
   Postupujeme obdobně jako výše.*)

   MFFF = Array[0 &, {Length[M]}];(*Implementace proměnné MFFF,
```

```
   je to pole samých nul délky stejné jako M.*)

   For[i = 1, i <= Length[M], i++,
    For[j = 1, j <= Length[lt], j++,
     If[PolynomialLCM[lt[[j]], M[[i]]] === M[[i]],
      A = M[[i]]/lt[[j]];
      MFFF = Union[MFFF, {A*GG[[j]]}];
      Break[]]]];
   (*Zkoumáme, zda existuje takové m', kde m=m'*lt. Porovnáváme tedy,
   zda nejmenší společný násobek (lt,M)===M. Pokud ano,
   tak takové m' existuje a spočítáme si ho a uložíme do proměnné A a \
do MFFF přidáme A*GG. Pokud ne, tak nepřidáváme nic.*)

   MNove = Expand[MFFF];(*Roznásobení MFFF*)

   Fkonecne = DeleteCases[Union[MNove, F], 0];(*Sjednotíme F,Mnove.
   Poté vymažeme nulové případy.*)

   Return[Fkonecne](*Vrací množinu polynomů v proměnné Fkonecne*)
   \
];

F4Reduction[LL_List, GG_List] :=
 Module[{FF, monomy, Fmat, Frowred, Fvysl, LTF, Fvyslplus, LTFvysl},
  (*Funkce F4Reduction.
  Vstupem je množina polynomů GG a množina čtveřic LL.
  Výstupem je množina polynomů Fvyslplus.*)

  FF = SymbolicPreprocessing[LL, GG];(*Zavolá algoritmus
  SymbolicPreprocessing*)

  monomy = MonomialList[
    DeleteDuplicates[Map[(If[#[[0]] === Times ,
          If[NumericQ[#[[1]]], #/#[[1]], #],
          If[NumericQ[#], 1, #]]) &,
       DeleteDuplicates[Flatten[FF /. Plus -> List]]]] /. List -> Plus];
  (*Vypíše všechny monomy vyskytující se v proměnné FF a seřadí je \
lexikograficky. Postup je uveden výše ve funkci SymbolicPreprocessing*)

    Fmat = Map[
    Total[Table[
       If[NumericQ[#[[i]]/monomy[[j]]], #[[i]]/monomy[[j]], 0], {i, 1,
         Length[#]}, {j, 1, Length[monomy]}]] &,
    Evaluate[MonomialList[FF]]];
  (*Plnění matice koeficienty u daných polynomů.
  Sloupce odpovídají jednotlivým monomům,
  řádky pak jednotlivým polynomům.
```

```
  Pokud Daný člen z FF podělený příslušným monomem je číslo(tzn.
  že se daný monom v polynomu vyskytuje), napíše tento koeficient,
  jinak napíše nulu.*)

  Frowred = RowReduce[Fmat];(*Řádkově schodovitý tvar matice*)

  Fvysl = DeleteCases[Frowred.monomy,
    0];(*Vypíše polynomy z řádkově schodovité matice.
  Vynásobí danou matici monomy a vypíše je bez nulových členů.*)

  LTF = DeleteDuplicates[
    Map[If[#[[0]] === Times , If[NumericQ[#[[1]]], #/#[[1]], #],
       If[NumericQ[#], 1, #]] &, MonomialList[FF][[All, 1]]]];
  (*Vypíše hlavní monomy z proměnné FF a pokud je některý monom shodný \
s jiným, tak ho vypíše jen jednou. Postup opět popsán výše.*)

  LTFvysl =
   DeleteDuplicates[
    Map[If[#[[0]] === Times , If[NumericQ[#[[1]]], #/#[[1]], #],
       If[NumericQ[#], 1, #]] &, MonomialList[Fvysl][[All, 1]]]];
  (*Vypíše hlavní monomy z proměnné Fvysl a pokud je některý monom \
shodný s jiným, tak ho vypíše jen jednou. Postup opět popsán výše.*)

   Fvyslplus =
   DeleteCases[
    Flatten[Table[
      If[Total[
         Table[If[LTFvysl[[j]] === LTF[[i]], 1, 0], {i, 1,
           Length[LTF]}]] == 0, Fvysl[[j]], 0], {j, 1,
       Length[LTFvysl]}]], 0];
  (*Zkoumáme, zda LT(Fvysl) je shodné s LT (F). Pokud ano,
  vypíše jedničku, jinak nulu. Pokud je celý řádek nulový,
  pak se daný člen v FF nevyskytuje a vypíšeme odpovídající polynom z \
Fvysl, jinak vypíšeme nulu. Nulové členy poté vymažeme.*)

  Return[Fvyslplus](*Vypíše výslednou množinu polynomů Fvyslplus*)
  ];
F4alg[Pol_List] :=
  Module[{lt, lcm, t, Par, L, F, d, G, P, Fplus, LTFplus, LTG},
   (*F4 algoritmus.
   Vstupem je množina polynomů a výstupem je Gröbnerova báze.*)

   lt = Map[MonomialList[#][[1]] &, Pol];
   lcm = Array[0 &, {Length[lt], Length[lt]}];
   t = Array[0 &, {Length[lt], Length[lt], Length[lt]}];
   Par = Array[0 &, {Length[lt], Length[lt]}];
   L = Array[0 &, {Length[lt], Length[lt]}];(*Inicializace proměnných*)
```

```
      For[i = 1, i <= Length[lt], i++,
    For[j = 1, j <= Length[lt], j++,
     If[i < j,
      lcm[[i, j]] =
       PolynomialLCM[lt[[i]],
        lt[[j]]];(*Spočítá nejmenší společný násobek dvou polynomů*)

          t[[i, j, i]] = lcm[[i, j]]/lt[[i]];(*Vypočítá členy t*)

         t[[i, j, j]] = lcm[[i, j]]/lt[[j]];
      Par[[i, j]] = {lcm[[i, j]], t[[i, j, i]], Pol[[i]],
        t[[i, j, j]],
        Pol[[j]]};(*Vytvoří pětice obsahující nejmenší společný \
násobek hl. členů dvou polynomů, člen t, polynom, člen t, polynom*)


      P = DeleteCases[Flatten[Par, 1],
        0];(*Vymaže nulové páry a poskládá pětice za sebe*)

      Null(*Vše děláme pouze pro i<j,
      abychom se nedostali k duplicitním pěticím*)
      ]
     ]
    ];
   G = Pol;(*Do proměnné G uložíme množinu zadaných polynomů*)

   While[! (P /. List -> Plus) === 0,(*Ukončující podmínka.
    Pokud P není nulové.*)

    L = P[[All, {2, 3, 4, 5}]];(*Vezme 2.-5. člen z P*)

    F = Flatten[
      Map[{#[[1]] #[[2]], #[[3]] #[[4]]} &,
       L]];(*V množině L vynásobí v každé podmnožině 2.3. a 4.5.
    člen a naskládá je za sebe do jedné množiny*)

    Fplus = F4Reduction[F, G];(*Zavolá algoritmus F4Reduction*)

    LTFplus =
     DeleteDuplicates[Map[
       If[#[[0]] === Times , If[NumericQ[#[[1]]], #/#[[1]], #], #] &,
       MonomialList[Fplus][[All, 1]]]];
    (*Spočítá hlavní monomy z polynomů,
    které jsou výstupem F4Reduction*)

    LTG = DeleteDuplicates[
```

```
      Map[If[#[[0]] === Times ,
         If[NumericQ[#[[1]]], #/#[[1]], #], #] &,
       MonomialList[G][[All, 1]]]];
    (*Spočítá hlavní monomy ze zadaných polynomů.*)

    P = Flatten[
      Expand[Simplify[
        Table[{PolynomialLCM[LTFplus[[i]], LTG[[j]]],
          PolynomialLCM[LTFplus[[i]], LTG[[j]]]/LTFplus[[i]],
          Fplus[[i]], PolynomialLCM[LTFplus[[i]], LTG[[j]]]/LTG[[j]],
          G[[j]]}, {i, 1, Length[LTFplus]}, {j, 1, Length[LTG]}]]], 1];
    (*Naplnění pětic*)

    G = Union[G, Fplus];(*Do G přidá nové polynomy z FPlus*)
    ];
   Return[G];(*Výsledkem je Gröbnerova báze G*)
   ];
End[];
Protect[SymbolicPreprocessing,F4Reduction,F4alg];

EndPackage[]
```

6.2.2. Lagrangeův interpolační polynom

```
#!/usr/bin/env python
# -*- coding: utf8 -*-


def multiply_poly(*args):
    # Funkce pro násobení několika polynomů. Vstupem jsou polynomy a výstupem
    # součin polynomů.Polynomy zapisujeme pomocí koeficientů dle jednotlivých
    # mocnin:[x^0, x^1, x^2, ..., x^n]
    """
    Multiplies n polynomials.
    @param *args: either single list of polynomials or polynomials as args.
    @return: args[0] * args[1] * ... * args[n]
    """
    if len(args) == 1:
        args = args[0]
    elif not args:
        return []
    res = [1]
    for number in args:
        res = _multiply_poly(res, number)
    return res
```

```
def _multiply_poly(poly1, poly2):
    # Funkce pro násobení dvou polynomů. Vstupem jsou polynomy a výstupem
    # součin polynomů. Polynomy zapisujeme pomocí koeficientů dle jednotlivých
    # mocnin:[x^0, x^1, x^2, ..., x^n]
    """
    Multiplies two polynomials.
    @param poly1: First polynom [x^0, x^1, x^2, ..., x^n]
    @param poly2: Second polynom [x^0, x^1, x^2, ..., x^n]
    @return: poly1 * poly2
    """
    res = [0, ] * (len(poly1) + len(poly2) - 1)
    for i in xrange(len(poly1)):
        for j in xrange(len(poly2)):
            idx = i + j
            res[idx] = sum_int(res[idx], multiply_int(poly1[i], poly2[j]))
    return res


def sum_poly(*args): #Funkce pro součet polynomů.
    """
    Sumarize polynomials.
    @param *args: either single list of polynomials or polynomials as args.
    @return: args[0] + arg[1] + ... + arg[n]
    """
    if len(args) == 1:
        args = args[0]
    elif len(args) < 2:
        return args
    res = [0] * max([len(_) for _ in args])
    for number in args:
        for idx in xrange(len(number)):
            res[idx] = sum_int(res[idx], number[idx])
    return res


def sum_int(num1, num2):
    # Funkce pro součet dvou čísel. Je zde zahrnuto, že -a=a, a+a=0, 0+a=a
    # a dále symetrie zobrazení. Je zde vypsána celá tabulka pro součet nad
    # daným polem.
    """
    Sum two numbers according to the mapping table.
    @param num1: First integer
    @param num2: Second integer
    @return: num1 + num2
    """
    # - a = a
    if num1 < 0:
```

```
        num1 = -num1
    if num2 < 0:
        num2 = -num2
    # a + a = 0
    if num1 == num2:
        return 0
    # symetrie
    if num1 > num2:
        tmp = num1
        num1 = num2
        num2 = tmp
    # 0 + a = a
    if num1 == 0:
        return num2
    mapping = {( 1,  2):  3, ( 1,  3):  2, ( 1,  4):  5, ( 1,  5):  4,
               ( 1,  6):  7, ( 1,  7):  6, ( 1,  8):  9, ( 1,  9):  8,
               ( 1, 10): 11, ( 1, 11): 10, ( 1, 12): 13, ( 1, 13): 12,
               ( 1, 14): 15, ( 1, 15): 14, ( 2,  3):  1, ( 2,  4):  6,
               ( 2,  5):  7, ( 2,  6):  4, ( 2,  7):  5, ( 2,  8): 10,
               ( 2,  9): 11, ( 2, 10):  8, ( 2, 11):  9, ( 2, 12): 14,
               ( 2, 13): 15, ( 2, 14): 12, ( 2, 15): 13, ( 3,  4):  7,
               ( 3,  5):  6, ( 3,  6):  5, ( 3,  7):  4, ( 3,  8): 11,
               ( 3,  9): 10, ( 3, 10):  9, ( 3, 11):  8, ( 3, 12): 15,
               ( 3, 13): 14, ( 3, 14): 13, ( 3, 15): 12, ( 4,  5):  1,
               ( 4,  6):  2, ( 4,  7):  3, ( 4,  8): 12, ( 4,  9): 13,
               ( 4, 10): 14, ( 4, 11): 15, ( 4, 12):  8, ( 4, 13):  9,
               ( 4, 14): 10, ( 4, 15): 11, ( 5,  6):  3, ( 5,  7):  2,
               ( 5,  8): 13, ( 5,  9): 12, ( 5, 10): 15, ( 5, 11): 14,
               ( 5, 12):  9, ( 5, 13):  8, ( 5, 14): 11, ( 5, 15): 10,
               ( 6,  7):  1, ( 6,  8): 14, ( 6,  9): 15, ( 6, 10): 12,
               ( 6, 11): 13, ( 6, 12): 10, ( 6, 13): 11, ( 6, 14):  8,
               ( 6, 15):  9, ( 7,  8): 15, ( 7,  9): 14, ( 7, 10): 13,
               ( 7, 11): 12, ( 7, 12): 11, ( 7, 13): 10, ( 7, 14):  9,
               ( 7, 15):  8, ( 8,  9):  1, ( 8, 10):  2, ( 8, 11):  3,
               ( 8, 12):  4, ( 8, 13):  5, ( 8, 14):  6, ( 8, 15):  7,
               ( 9, 10):  3, ( 9, 11):  2, ( 9, 12):  5, ( 9, 13):  4,
               ( 9, 14):  7, ( 9, 15):  6, (10, 11):  1, (10, 12):  6,
               (10, 13):  7, (10, 14):  4, (10, 15):  5, (11, 12):  7,
               (11, 13):  6, (11, 14):  5, (11, 15):  4, (12, 13):  1,
               (12, 14):  2, (12, 15):  3, (13, 14):  3, (13, 15):  2,
               (14, 15):  1,
              }
    return mapping[(num1, num2)]


def multiply_int(num1, num2):
    # Funkce pro násobení dvou čísel. Opět zahrnuto, že -a=a, 0*a=0, 1*a=a
```

```
    # a symetrie. Je zde vypsána tabulka pro násobení nad daným polem.
    """
    Multiplies two numbers according to the mapping table.
    @param num1: First integer
    @param num2: Second integer
    @return: num1 * num2
    """
    # - a = a
    if num1 < 0:
        num1 = -num1
    if num2 < 0:
        num2 = -num2
    # symetrie
    if num1 > num2:
        tmp = num1
        num1 = num2
        num2 = tmp
    # 0 * a = 0
    if num1 == 0:
        return 0
    # 1 * a = a
    if num1 == 1:
        return num2
    mapping = {( 2,  2):  3, ( 2,  3):  1, ( 2,  4):  8, ( 2,  5): 10,
               ( 2,  6): 11, ( 2,  7):  9, ( 2,  8): 12, ( 2,  9): 14,
               ( 2, 10): 15, ( 2, 11): 13, ( 2, 12):  4, ( 2, 13):  6,
               ( 2, 14):  7, ( 2, 15):  5, ( 3,  3):  2, ( 3,  4): 12,
               ( 3,  5): 15, ( 3,  6): 13, ( 3,  7): 14, ( 3,  8):  4,
               ( 3,  9):  7, ( 3, 10):  5, ( 3, 11):  6, ( 3, 12):  8,
               ( 3, 13): 11, ( 3, 14):  9, ( 3, 15): 10, ( 4,  4):  7,
               ( 4,  5):  3, ( 4,  6): 15, ( 4,  7): 11, ( 4,  8):  9,
               ( 4,  9): 13, ( 4, 10):  1, ( 4, 11):  5, ( 4, 12): 14,
               ( 4, 13): 10, ( 4, 14):  6, ( 4, 15):  2, ( 5,  5):  6,
               ( 5,  6):  9, ( 5,  7): 12, ( 5,  8):  1, ( 5,  9):  4,
               ( 5, 10): 11, ( 5, 11): 14, ( 5, 12):  2, ( 5, 13):  7,
               ( 5, 14):  8, ( 5, 15): 13, ( 6,  6):  4, ( 6,  7):  2,
               ( 6,  8):  5, ( 6,  9):  3, ( 6, 10): 14, ( 6, 11):  8,
               ( 6, 12): 10, ( 6, 13): 12, ( 6, 14):  1, ( 6, 15):  7,
               ( 7,  7):  5, ( 7,  8): 13, ( 7,  9): 10, ( 7, 10):  4,
               ( 7, 11):  3, ( 7, 12):  6, ( 7, 13):  1, ( 7, 14): 15,
               ( 7, 15):  8, ( 8,  8): 14, ( 8,  9):  6, ( 8, 10):  2,
               ( 8, 11): 10, ( 8, 12):  7, ( 8, 13): 15, ( 8, 14): 11,
               ( 8, 15):  3, ( 9,  9): 15, ( 9, 10):  8, ( 9, 11):  1,
               ( 9, 12): 11, ( 9, 13):  2, ( 9, 14):  5, ( 9, 15): 12,
               (10, 10): 13, (10, 11):  7, (10, 12):  3, (10, 13):  9,
               (10, 14): 12, (10, 15):  6, (11, 11): 12, (11, 12): 15,
               (11, 13):  4, (11, 14):  2, (11, 15):  9, (12, 12):  9,
```

```
                (12, 13):  5, (12, 14): 13, (12, 15):  1, (13, 13):  8,
                (13, 14):  3, (13, 15): 14, (14, 14): 10, (14, 15):  4,
                (15, 15): 11,
               }
    return mapping[(num1, num2)]

def divide_int(num1, num2):
    # Funkce pro dělení dvou čísel. Zkoumá, jaké číslo musíme vynásobit s num2,
    # abychom dostali num1.
    for i in xrange(16):
        if multiply_int(num2, i) == num1:
            return i

def print_poly(poly):  #Funkce pro výpis polynomu pomocí x^d
    """ String output """
    out = ""
    for i in xrange(len(poly)):
        if poly[i] == 0:
            continue
        elif poly[i] == 1:
            out += "x^%d + " % i
        else:
            out += "%sx^%d + " % (poly[i], i)
    if out:
        out = out[:-3]
    return out


def calculate_lagrange():
    # Hlavní funkce programu. Slouží k výpočtu Lagrangeova interpolačního
    # polynomu nad daným polem.
    """
    Calculates lagrange according to the mapping
    """

    mapping = [9, 11, 11, 11, 13, 4, 10, 1, 13, 1, 4, 10, 13, 10, 1, 4]
    # Funkční hodnoty v jednotlivých bodech.
    length = len(mapping)
    # Polynom x^3 -2x + 1 přepíšeme do programu jako [1, -2, 0, 1]
    ls_u = [] #proměnná typu list
    ls_d = []
    for i in xrange(length):
        # (x-1)(x-2)(x-3)...(x-$posledni) if i!=idx
        numerator = multiply_poly([[_, 1] for _ in xrange(length) if _ != i])
        # (idx-0)(idx-1)...(idx-$posledni) if i!=idx
        denominator = 1
        for _ in (sum_int(i, _) for _ in xrange(length) if _ != i):
```

```
            denominator = multiply_int(denominator, _)
        prefix = "l%-2d = " % i
        # l:vypíše l, d:integer, -2: zarovnání doleva na 2 znaky
        _prefix = " " * len(prefix)  # násobí mezeru- pro lepší výstup
        print "%s%s" % (_prefix, numerator)
        print "%s%s" % (prefix, '-' * (max(len(str(numerator)),
                                          len(str(denominator)))))
        print "%s%s" % (_prefix, denominator)
        ls_u.append(numerator)
        ls_d.append(denominator)


    # daný polynom vynásobí příslušnou funkční hodnotou: l$idx * mapping[$idx]
    Ls = [multiply_poly([divide_int(mapping[i], ls_d[i])], ls_u[i])
          for i in xrange(length)]
    print "Ls = %s" % Ls

    # součet všech Ls. Tím získáme výsledný Lagrangeův polynom
    L = print_poly(sum_poly(Ls))
    print "L = %s" % L


if __name__ == "__main__":
    calculate_lagrange()
```