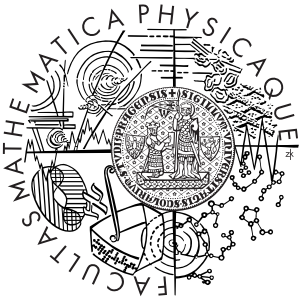
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Jan Seidl

Řešení soustav rovnic nad komutativními okruhy

Katedra Algebry

Vedoucí bakalářské práce: Mgr. Jan Šťovíček, Ph.D.

Studijní program: Matematika

Studijní obor: Matematické metody informační bezpečnosti

Praha 2012

Poděkování

Rád bych na tomto místě poděkoval svému školiteli, Mgr. Janu Šťovíčkovi, Ph.D., za jeho vstřícnost, ochotu a odborné rady.

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Praze dne Podpis:

Název práce: Řešení soustav rovnic nad komutativními okruhy

Autor: Jan Seidl

Katedra: Katedra algebry

Vedoucí bakalářské práce: Mgr. Jan Šťovíček, Ph.D., Katedra algebry

Abstrakt: Předmětem této práce je nabídnout algoritmus, jakým se dají řešit soustavy lineárních rovnic Ax=b nad okruhy hlavních ideálů. Dokážeme, že ke každé nenulové matici nad okruhem hlavních ideálů existuje její Smithův tvar. Užitím Smithova tvaru převedeme danou soustavu do jednoduché diagonální podoby a ukážeme, jak z řešení soustavy v této diagonální podobě lze získat řešení původní soustavy. Celý postup demonstrujeme na příkladech pro okruhy Z, Zm a Q[x]. Následně předvedeme, jak je možné algoritmus pro jednotlivé okruhy implementovat v programu Mathematica.

Práce by měla také poskytnou postup, podle kterého by nemělo být obtížné modifikovat algoritmus tak, aby bylo možné získat řešení soustav i pro jiné okruhy.

Klíčová slova: algoritmus, okruh, Smithův tvar, Mathematica

Title: Solving systems of equations over commutative rings

Author: Jan Seidl

Department: The Department of Algebra

Supervisor: Mgr. Jan Šťovíček, Ph.D., The Department of Algebra

Abstract: The object of this work is to offer algorithm how can be solved systems of linear equations Ax=b over principal ideal rings. We prove that for every nonzero matrix over principal ideal rings there exists its Smith form. Using Smith form we transform the system of equations to simple diagonal form and we show how we can obtain the solution of the original system from its diagonal form. Whole procedure we demonstrate by the examples over Z, Zm and Q[x]. Thereafter we show how is possible to implement the algorithm for these rings by using software Mathematica.

The work should provide procedure according to which shold not be difficult to modify algorithm to gain solution over another rings.

Keywords: algorithm, ring, Smith form, Mathematica

Obsah

Úvod

Řešení soustav lineárních rovnic patří mezi staré matematické problémy. Pro tělesa byla již navržena řada metod, jakým způsobem lze získat řešení. Příkladem může být Gaussova eliminační metoda, metoda nejmenších čtverců či řešení pomocí inverzní matice. S rozvojem abstraktní algebry, zejména v období konce 19. a počátkem 20. století, vyvstala potřeba řešit soustavy rovnic i nad jinými obory. Dnes se proto lze setkat s řešením soustav lineárních rovnic nad různými obory nejen v lineární algebře, ale také v odvětvích aplikované matematiky, jakými jsou například lineární kryptoanalýza či lineární programování.

Cílem této práce je ukázat, jakým způsobem se dají řešit soustavy lineárních rovnic nad okruhy hlavních ideálů a tento postup implementovat v programu Mathematica. Máme-li tedy zadanou nenulovou soustavu rovnic v maticovém zápisu Ax=b, pak nad okruhy hlavních ideálů lze matici A převést do tzv. Smithova normálního tvaru, který je diagonální maticí. Všechny operace na matici A jsou zaznamenávány do pomocných matic, které následně umožňují vhodně upravit pravou stranu rovnosti a převést tak zadanou soustavu do diagonální podoby, ve které je již soustava snadno řešitelná. Ze získaného řešení lze pak opět využitím pomocných matic dostat řešení původní soustavy.

V prvních dvou kapitolách jsou uvedeny základní definice a nezbytná tvrzení z teorie okruhů a modulů. Platnost těchto tvrzení je pak předpokladem pro třetí kapitolu, kde se dokazuje věta o existence Smithova normálního tvaru pro každou matici nad okruhem hlavních ideálů. Součástí důkazu této věty je také jeho konstruktivní verze, která je zároveň zobecněným postupem, jak ze zadané matice získat její Smithův normální tvar. Následující kapitoly jsou pak rozdělené podle konkrétních okruhů, nad kterými se zadaná soustava rovnic řeší. V kapitole čtyři se řeší soustava rovnic nad okruhem celých čísel. V kapitole pět je řešena soustava rovnic nad okruhem polynomů nad racionálními čísly. Obě tyto kapitoly jsou rozděleny do tří částí, v prvních částích je slovně popsán algoritmus nad konkrétním okruhem, následuje ukázka algoritmů na příkladu a následně jejich možná implementace v programu Mathematica. V šesté kapitole je řešena soustava rovnic nad okruhem Z_m. Tato kapitola je rozdělena do tří podkapitol v závislosti na m. Jednotlivé podkapitoly obsahují slovní popis algoritmu, následně ukázku na příkladu a možnou implementaci algoritmu v programu Mathematica.

Kapitola 1

Okruhy

Definice 1 *Okruh $R = (R, +, *, 1)$ je množina R s dvěma binárními operacemi (sčítáním a násobením), taková, že*
1. (R, +) je komutativní grupa vzhledem na sčítání;
*2. (R, *) je monoid vzhledem na násobení;*
3. násobení je distributivní (oboustranné) vzhledem na sčítání.

Komutativní okruh je okruh, ve kterém je násobení komutativní.
[6, podkapitola 4.1]

Definice 2 *Prvky a,b komutativního okruhu K jsou dělitelé nuly v K, jestliže $a \neq 0, b \neq 0$ a $ab = 0$. Oborem integrity je pak netriviální komutativní okruh, který nemá dělitele nuly.*
[6, podkapitola 4.4]

Definice 3 *(Oboustranný) ideál A okruhu R je neprázdná podmnožina A množiny R, pro kterou platí:*
1. $\alpha_1, \alpha_2 \in A \Rightarrow \alpha_1 - \alpha_2 \in A$,
2. $r \in R$ a $\alpha \in A \Rightarrow r\alpha \in A$ a $\alpha r \in A$.

Jestliže K je komutativní okruh, pak množina $kb = \{kb; k \in K\}$ všech "násobků"pevně zvoleného prvku $b \in K$ je ideál okruhu K, který budeme značit kb, anebo jednoduše (b) a nazývat ho *hlavním* ideálem.
Nechť D je obor integrity. Hlavní ideály oboru integrity D úzce souvisí s vlastnostmi dělení v D. Pokud $(a) \subset (b)$, potom $a = bc$ pro nějaké $c \in D$ a tedy $b \mid a$ v D. Obráceně, z $b \mid a$ vyplývá $(a) \subset (b)$. Podobně $(a) = (b)$ právě tehdy, když a a b jsou asociované prvky z D, speciálně $(a) = D$ právě když a je dělitelem jednotky v D.
[6, podkapitola 4.3]

Definice 4 *Okruh hlavních ideálů D je okruh, jehož každý ideál je hlavní ideál.*

Tvrzení 1 *V okruhu hlavních ideálů D je každá neklesající posloupnost $C_1 \subset C_2 \subset ... \subset C_k \subset ...$ ideálů počínající od určitého členu konstantní, t.j. existuje index m takový, že $C_m = C_{m+1} = ...$*

Důkaz: Protože každé C_k je ideál, sjednocení C všech množin C_k je uzavřené vzhledem na sčítání a násobení a je to tedy ideál okruhu D, přičemž však $C = (c)$ pro nějaký prvek c. Tento prvek musí patřit alespoň do jednoho z ideálů C_k, například do C_m. Potom C_m se už musí rovnat celému ideálu C, a tak $C_m = C_{m+1} = ...$
[6, podkapitola 4.9]

Věta 1 *V okruhu hlavních ideálů D má každá dvojice nenulových prvků a, b největší společný dělitel d, který je možný vyjádřit jako lineární kombinaci $d = sa + tb$ prvků a, b s vhodnými koeficienty s, $t \in D$.*

Důkaz: Množina $C = (a, b)$ všech možných lineárních kombinací $xa + yb$, kde $x, y \in D$, je ideál okruhu D, a tedy hlavní ideál (d) pro nějaké d tvaru $sa + tb$. Všechny prvky množiny C, a tedy i dané prvky $a = 1a + 0b$ a $b = 0a + 1b$ jsou násobky tohoto d. Na druhou stranu, jestliže c je nějaký společný dělitel prvků a, b, tak, že $a = ác$ a $b = \acute{b}c$ pro vhodné $á, \acute{b}$, tak $d = sa + tb = (sá + t\acute{b})c$ je násobek prvku c. To znamená, že d je největší společný dělitel a, b.

Důsledek 1 *Jestliže je p ireducibilní prvek okruhu hlavních ideálů D, potom*

$$p \mid ab \Rightarrow p \mid a \text{ anebo } p \mid b, \ (a, b \in D).$$

Důkaz: Ireducibilní prvek p je takový prvek, který je dělitelný jen dělitelmi jednotky a prvky asociovanými s p. Tedy $NSD(p, a)$ je roven buď p anebo 1. V prvním případě $p \mid a$, ve druhém případě dle Věty 1 můžeme vyjádřit $NSD(p, a) = 1 = sa + tp$, $s, t \in D$ a tedy $b = b.1 = sab + tbp$. Podle předpokladu p dělí součin ab, dělí tedy oba sčítance na pravé straně rovnosti a tedy dělí i prvek b.

Věta 2 *Každý prvek $a \neq 0$ okruhu hlavních ideálů je buď dělitel jednotky, anebo součin $a = p_1 \cdots p_m$ ireducibilních prvků p_i. Jestliže také platí, že $a = q_1 \cdots q_n$ a q_i jsou ireducibilní, tak $n = m$ a existuje permutace σ indexů $\{1, ..., n\}$ taková, že každé $q_{\sigma i}$ je asociované s p_i.*

Důkaz: Nechť D je okruh hlavních ideálů. Předpokládejme, že nějaký prvek $a \in D, a \neq 0$, není dělitel jednotky a nemá rozklad na ireducibilní činitele. Potom a není ireducibilní v D, můžeme ho tedy napsat jako součin $a = a_1 b_1$ dvou vlastních dělitelů. Kdyby pro a_1 i b_1 existoval rozklad na ireducibilní činitele, existoval by také rozklad i pro a. Tedy pro jeden z prvků a_1, b_1, například pro a_1, neexistuje roklad na ireducibilní činitele. Podobně jako předtím dostáváme $a_1 = a_2 b_2$ a pro a_2 neexistuje rozklad, takto můžeme pokračovat dále. Dostaneme posloupnost $a_1, a_2, ..., a_k$ prvků. Protože každé a_{k+1} je vlastním dělitelem prvku a_k, dostaneme nekonečně rostoucí posloupnost $(a) \subset (a_1) \subset (a_2) \subset ...$ hlavních ideálů okruhu D, což je ve sporu s Tvrzením 1.

Nyní zbývá dokázat, že pokud $a \neq 0$ má rozklad s m ireducibilními činiteli, tak je tento rozklad jednoznačný. Dokážeme to indukcí podle m. Jestliže $m = 1$, tak $a = p$ je ireducibilní prvek a má zřejmě jen jediný rozklad. Předpokládejme, že každý prvek, který je součinem m ireducibilních činitelů, má jednonačný rozklad a uvažujme rozklady $p_1 p_2 \cdots p_m p_{m+1} = a = q_1 q_2 \cdots q_n$. Ireducibilní prvek p_{m+1} dělí levou stranu, tedy i pravou stranu, t.j., dle Důsledku 1 dělí jeden z činitelů q_i na

pravé straně. Oba tyto prvky jsou ireducibilní, a proto p_{m+1} a q_i jsou asociované: $q_i = up_{m+1}$. Rovnost $p_1p_2 \cdots p_mp_{m+1} = a = q_1q_2 \cdots q_n$ můžeme vykrátit prvkem p_{m+1} a použít indukční předpoklad.
[6, podkapitola 4.10]

Definice 5 *Euklidovskou normou na oboru R rozumíme zobrazení*

$$\varphi : R \to N \cup \{0\}$$

splňující
1) $\varphi(0) = (0)$;
2) pokud $a \mid b \neq 0$, pak $\varphi(a) \leq \varphi(b)$;
3) pro všechna $a, b \in R$, $b \neq 0$, existují $q, r \in R$ taková, že $a = bq + r$ a $\varphi(r) < \varphi(b)$.

Definice 6 *Obor R se nazývá Euklidovský, pokud na něm existuje Euklidovská norma.*
[8, podkapitola 7.1]

Věta 3 *V Euklidovských oborech je každý ideál hlavní.*

Důkaz: Buď I ideál v Euklidovském oboru R. Je-li $I = \{0\}$, pak $I = 0R$. V opačném případě označme a takový prvek ideálu I, který má nejmeší nenulovou Euklidovskou normu (libovolný z nich, je-li jich více). Dokážeme, že $I = aR$. Zřejmě $ar \subseteq I$, pro spor tedy předpokládejme, že esistuje nějaký prvek $b \in I \setminus aR$. Zvolme q, r splňující $b = aq + r$ a $\varphi(r) < \varphi(a)$. Samozřejmě $r \neq 0$, protože b není dělitelné a, a tedy $0 < \varphi(r) < \varphi(a)$. Avšak $r = b - aq \in I$ neboť $b \in I$ a $aq \in I$, což je ale spor s výběrem a jako prvku I s nejmenší kladnou normou.

Tělesa jsou Euklidovské obory. Euklidovskou normou je např. zobrazení $\varphi(0) = 0$ a $\varphi(a) = 1$ pro všechna $a \neq 0$.
Obor celých čísel Z je Euklidovský. Euklidovskou normou je absolutní hodnota, t.j., $\varphi(a) =\mid a \mid$.
Obor $F[x]$ je Euklidovský, pokud F je těleso. Euklidovskou normou je $\varphi(f) = 1 + deg(f)$, kde $deg(f)$ značí stupeň polynomu f.
[8, podkapitola 7.2]

Definice 7 *Řekneme, že dva ideály A a B okruhu R jsou komaximální, pokud $A+B=R$.*

Věta 4 *Nechť $A_1, A_2, ..., A_k$ jsou ideály okruhu R a zobrazení $R \to R/A_1 \times R/A_2 \times \cdots \times R/A_k$ definované $r \to (r + A_1, r + A_2, ..., r + A_k)$ je okruhový homomorfismus s jádrem $A_1 \cap A_2 \cap \cdots A_k$. Jestliže pro každé $i, j \in \{1, 2, ..., k\}$, kde $i \neq j$, jsou ideály A_i a A_j komaximální, potom toto zobrazení je surjektivní a $A_1 \cap A_2 \cap \cdots \cap A_k = A_1A_2 \cdots A_k$, tedy $R/(A_1A_2 \cdots A_k) = R/(A_1 \cap A_2 \cap \cdots \cap A_k) \cong R/A_1 \times R/A_2 \times \cdots \times R/A_k$.*

Důkaz: Indukcí podle počtu ideálů k. Nechť nejprve $k = 2$, tedy $A = A_1$ a $B = A_2$. Uvažujme zobrazení $\varphi : R \to R/A \times R/B$ definované $\varphi(r) = (r \ mod \ A, r \ mod \ B)$, kde $mod \ A$ znamené rozkladovou třídu faktorokruhu R/A

obsahující r. Zobrazení φ je homomorfismus, jádro φ se skládá z prvků $r \in R$, obsažných v A i B, t.j., $A \cap B$. Zbývá tedy dokázat, že pokud jsou A i B komaximální, potom φ je surjektivní a $A \cap B = AB$. Jelikož $A + B = R$, potom existují prvky $x \in A, y \in B$ splňující $x + y = 1$. Nechť např. $x \in A$ a $x = 1 - y \in 1 + B$, potom tedy $\varphi(x) = (0, 1)$ a $\varphi(y) = (1, 0)$. Jestliže $(r_1 \ mod \ A, r_2 \ mod \ B)$ je libovolný prvek z $R/A \times R/B$, potom se na tento prvek zobrazí prvek $r_2x + r_1y$, neboť

$$\varphi(r_2x + r_1y) = \varphi(r_2)\varphi(x) + \varphi(r_1)\varphi(y) =$$
$$(r_2 \ mod \ A, r_2 \ mod \ B)(0, 1) + (r_1 \ mod \ A, r_1 \ mod \ B)(1, 0) =$$
$$(0, r_2 \ mod \ B) + (r_1 \ mod \ A, 0) = (r_1 \ mod \ A, r_2 \ mod \ B).$$

Tedy zobrazení φ je surjektivní. Ideál AB je vždy obsažen v průniku $A \cap B$ a předpokládejme, že A, B jsou komaximální, potom pro libovolný prvek $c \in A \cap B$ platí $c = c1 = cx + cy \in AB$. Z toho plyne i opačná inkluze $A \cap B \subseteq AB$. Tím je hotov důkaz pro $k = 2$.
V obecném případě nechť ideály A_1 a $A_2 \cdots A_k$ jsou komaximální, potom využijme případ pro dva ideály, kde $A = A_1$ a $B = A_2 \cdots A_k$. Dle předchozího pro každé $i \in \{2, 3, ..., k\}$, existují prvky $x_i \in A_1$ a $y_i \in A_i$ splňující $x_i + y_i = 1$. Jelikož $x_i + y_i \equiv y_i \ mod \ A_1$, plyne z toho, že $1 = (x_2 + y_2) \dots (x_k + y_k)$ je prvek z $A_1 + (A_2 \cdots A_k)$.

Důsledek 2 *Nechť n je kladné celé číslo a $p_1^{\alpha_1} p_2^{\alpha_2} \dots p_k^{\alpha_k}$ jeho rozklad na součin mocnin různých prvočísel. Potom pro okruh Z_n platí*

$$Z_n \cong Z_{p_1^{\alpha_1}} \times Z_{p_2^{\alpha 2}} \times \dots \times Z_{p_k^{\alpha_k}}.$$

[2, podkapitola 7.6]

Kapitola 2

Moduly

Definice 8 *Nechť R je okruh. R-modul je aditivní komutativní grupa spolu s funkcí $R \times A \to A$, označovanou $(\omega, \alpha) \to \omega\alpha$, která splňuje následující axiomy pro všechny $\omega, \lambda \in R$ a $a, b \in A$:*
1) $\omega(a+b) = \omega a + \omega b$,
2) $(\omega + \lambda)a = \omega a + \lambda b$,
3) $(\omega\lambda)a = \omega(\lambda a)$,
4) $1a = a$.

Takto definovaný modul je *levý* modul, protože při tvoření λa skalár λ píšeme vlevo od a.
[6, podkapitola 6.1]

Definice 9 *Nechť R je okruh a M je R-modul. Potom R-podmodul N R-modulu M je podmnožina N množiny M, která je uzavřená na sčítání a operace prvků okruhu, t.j., $rn \in N$, pro všechna $r \in R$, $n \in N$.*
[2, podkapitola 10.1]

Definice 10 *Nechť M je R-modul a nechť $N_1, ..., N_n$ jsou podmoduly M.*

1) Součet $N_1, ..., N_n$ je množina všech konečných součtů prvků z množin N_i: $\{a_1 + a_2 + \cdots + a_n \mid a_i \in N_i$ pro všechna $i\}$. Označme tento součet jako $N_1 + \cdots + N_n$.

2) Pro libovolnou podmnožinu A množiny M nechť
$RA = \{r_1a_1 + r_2a_2 + \cdots + r_ma_m \mid r_1$, ..., $r_m \in R$, a_1, ..., $a_m \in A$, $m \in Z^+\}$
($RA = \{0\}$ jestliže $A = \emptyset$). Jestliže A je konečná množina $\{a_1, a_2, ..., a_n\}$, budeme psát $Ra_1 + Ra_2 + \cdots + Ra_n$ pro RA. RA nazveme podmodul M generovaný A. Jestliže N je podmodul M (možno $N = M$) a $N = RA$, pro nějakou podmnožinu A z M, nazveme A množonou generátorů nebo generující množinou N a říkáme, že N je generovaný A.

3) Podmodul N modulu M (možno $N = M$) je konečně generovaný, jestliže zde existuje konečná množina A z M, že $N = RA$, neboli, že N je generovaný nějakou konečnou podmnožinou.

4) Podmodul N modulu M (možno $N = M$) je cyklický, jestliže existuje prvek $a \in M$, že $N = Ra$, t.j., N je generován jedním prvkem: $N = Ra = \{ra \mid r \in R\}$.

Podmodul N R-modulu M může mít různé generující množiny. Jestliže je N konečně generovaný, pak existuje nejmenší kladné celé číslo d, že N je generovaný počtem d prvků. Libovolná množina obsahující d prvků bude nazývána *minimální množinou generátorů* N.
[2, podkapitola 10.3]

Definice 11 *Nechť R je komutativní okruh. R-modul M je Noetherovský R-modul, jestliže neexistuje nekonečný rostoucí řetězec jeho podmodulů, t.j., pokud $M_1 \subseteq M_2 \subseteq M_3 \subseteq \cdots$ je rostoucí řetězec podmodulů modulu M, potom existuje kladné celé číslo m, že pro všechna $k \geq m, M_k = M_m$.*

Definice 12 *Okruh R je Noetherovský, jestliže je Noetherovský jako modul sám nad sebou, t.j., jestliže neobsahuje žádný nekonečný řetězec ideálů v R.*

Věta 5 *Nechť R je okruh a M modul nad tímto okruhem. Pak následující tvrzení jsou ekvivalentní:*
1) M je Noetherovský R-modul.
2) Každá neprázdná množina podmodulů M obsahuje maximální prvek vzhledem k inkluzi.
3) Každý podmodul modulu M je konečně generovaný.

Důkaz:
(1) $\Rightarrow$ (2)
Předpokládáme, že M je Noetherovský a nechť Σ je nějaká neprázdná množina podmodulů modulu M. Vyberme $M_1 \in \Sigma$. Jestliže M_1 je maximální prvek Σ, pak implikace platí, takže předpokládejme, že M_1 není maximální. Potom zde existuje $M_2 \in \Sigma$, pro který platí $M_1 \subset M_2$. Jestliže M_2 je maximílní v Σ, pak tvrzení platí, jinak opět existuje $M_3 \in \Sigma$, že $M_2 \subset M_3$. Pokud bychom takto pokračovali do nekonečna, pak by to byl spor s tím, že R je Noetherovský.
(2) $\Rightarrow$ (3)
Nechť N je podmodul modulu M. Nechť Σ je množina všech konečně generovaných podmodulů modulu N. Jelikož $\{0\} \in \Sigma$, tato množina není prázdná. Dle tvrzení (2) Σ obsahuje maximální prvek $\acute{N}$. Jestliže $\acute{N} \neq N$, nechť $x \in N - \acute{N}$. Jelikož $\acute{N} \in \Sigma$, potom dle předpokladu je $\acute{N}$ konečně generovaný, tedy také podmodul generovaný $\acute{N}$ a x je konečně generovaný. To je ale ve sporu s maximalitou $\acute{N}$, tedy $N = \acute{N}$ je konečně generovaný.
(3) $\Rightarrow$ (1)
Nechť $M_1 \subseteq M_2 \subseteq M_3 ...$ je řetězec podmodulů modulu M. Nechť $N = \bigcup_{i=1}^{\infty} M_i$, pak N je také podmodul a dle tvrzení (3) konečně generovaný, řekněme prvky $a_1, a_2, ..., a_n$. Jelikož $a_i \in N$ pro všechna i, pak každé a_i leží v jednom z modulů v řetězci, řekněme v M_{ji}. Nechť $m = max\{j_1, j_2, ..., j_n\}$. Potom $a_i \in M_m$ pro všechna i, takže modul, který generují je obsažen v M_n, t.j., $N \subseteq M_m$. To implikuje $M_m = N = M_k$ pro všechna $k \geq m$.

Důsledek 3 *Jestliže R je obor hlavních ideálů, potom každá neprázdná množina ideálů okruhu R obsahuje maximální prvek a R je Noetherovský okruh.*
[2, podkapitola 12.1]

Definice 13 *Nechť $M_1, M_2, ..., M_k$ je posloupnost R-modulů. Potom množina k-tic $(m_1, m_2, ..., m_k)$, kde $m_i \in M_i$ spolu s operacemi sčítání a násobení prvků z R definovaných člen po členu, je nazývána (vnějším) direktním součtem $M_1, M_2, ..., M_k$, značeným $M_1 \oplus M_2 \oplus \ldots \oplus M_k$.*
[2, podkapitola 10.3]

Věta 6 *Nechť M je modul obsahující podmoduly $M_1, M_2, ..., M_n$ s následujícími vlastnostmi:*
1) $M = M_1 + M_2 + \ldots + M_n$,
2) pro každé i, $1 \leq i \leq n$: $M_i \cap (M_1 + \ldots + M_{i-1} + M_{i+1} + \ldots + M_n) = 0$.
Potom zobrazení

$$\nu : (x_1, ..., x_n) \to \sum_1^n x_i$$

je isomorfismus mezi $\oplus M_i$ a M. Naopak, pro $\oplus M_i$ nechť

$$\acute{M}_i = \{(0, ..., 0, \acute{x}_i, 0, ..., 0) \mid x_i \in M_i\}.$$

Potom jsou $\acute{M}_i$ podmoduly $\oplus M_i$ splňující podmínky 1), 2) a jsou isomorfní M_i.

Důkaz: Předpokládejme, že podmoduly M_i z M splňují podmínky 1), 2) a nechť ν je zobrazení definované $\nu : (x_1, ..., x_n) \to \sum_1^n x_i$. Jelikož platí

$$\nu(x_1 + y_1, ..., x_n + y_n) = \sum_1^n \nu(x_i + y_i) = \sum_1^n \nu(x_i) + \sum_1^n \nu(y_i) =$$
$$= \nu(x_1, ..., x_n) + \nu(y_1, ..., y_n)$$
a
$$\nu(ax_1, ..., ax_n) = \sum_1^n \nu(ax_i) = \sum_1^n a\nu(x_i) = a\sum_1^n \nu(x_i),$$

pak ν je homomorfismus z $\oplus M_i$ do M. Nyní ukážeme, že ν je surjektivní. Pro libovolný prvek $x \in M$ můžeme psát $x = \sum x_i,\ x_i \in M_i$, jelikož $M = \sum M_i$ dle podmínky 1) a $\sum M_i$ je množina prvků tvaru $\sum x_i,\ x_i \in M$. Potom $\nu(x_1, ..., x_n) = \sum x_i = x$.
Nyní ukážeme, že ν je také injektivní, t.j., stačí ukázat, že je-li $\nu(x_1, ..., x_n) = \sum_1^n x_i = 0$, potom pro všechna i platí $x_i = 0$. To je patrné z podmínky 2), neboť je-li $\sum_1^n x_i = 0$, pak $-x_i = \sum_{j \neq i} x_j$, odtud dostáváme, že $x_i \in M_i \cap (\sum_{j \neq i} M_j) = 0$.
Naopak, uvažujme $\oplus M_i$. Je zřejmé, že zobrazení $\nu_i : x_i \to (0, ..., 0, x_i, 0, ..., 0)$ je monomorfismus z M_i do M. Obrazem je $\acute{M}_i$, tedy $\acute{M}_i$ je podmodul M isomorfní M_i. Jelikož

$$(x_1, 0, ..., 0) + (0, x_2, 0, ..., 0) + \ldots + (0, ..., 0, x_n) = (x_1, x_2, ..., x_n),$$

pak je spněna podmínka 1) pro podmoduly $\acute{M}_i$ z M. Jelikož $\sum_{j \neq i} \acute{M}_j$ je množina prvků tvaru $(x_1, ..., x_{i-1}, 0, x_{i+1}, ..., x_n)$, platí tedy i podmínka 2).
[5, podkapitola 3.4]

Definice 14 *R-modul F se nazývá volný na podmnožině A množiny F, jestliže pro každý nenulový prvek x z F existují jednoznačně určené nenulové prvky* $r_1, r_2, ..., r_n$ *z R a prvky* $a_1, a_2, ..., a_n$ *v A, pro které platí, že* $x = r_1a_1 + r_2a_2 + \cdots + r_na_n$*, pro nějaké* $n \in Z^+$*. Potom říkáme, že A je bází nebo množinou volných generátorů F. Jestliže R je komutativní okruh, pak mohutnost A se nazývá stupněm F.*
[2, podkapitola 10.3]

O tom, že je *stupeň* volného modulu dobře definován, říká následující Věta 7. Existují totiž nekomutativní okruhy, pro které následující implikace neplatí. Více lze nalézt v [1, podkapitola 2.8].

Věta 7 *Jestliže R je komutativní okruh, potom* $R^{(m)} \cong R^{(n)}$ *implikuje* $m = n$.

Důkaz: Jestliže na danou větu nahlédneme z pohledu volných modulů, pak je tvrzení ekvivalentní: jestlliže je M volný modul nad komutativním okruhem R a M má báze s počtem prvků m a n, pak $m = n$. Tedy nechť $\{e_i \mid 1 \leq i \leq n\}, \{f_j \mid 1 \leq j \leq m\}$ jsou báze M. Potom máme

$$f_j = \sum_1^n a_{ji}e_i, \qquad e_i = \sum_1^m b_{ij}f_j,$$

kde $a_{ji}, b_{ij} \in R$. Substitucí dostaneme

$$f_j = \sum_{i=1,\acute{j}=1}^{n,m} a_{ji}b_{i\acute{j}}f_{\acute{j}}$$

$$e_i = \sum_{j=1,\acute{i}=1}^{m,n} b_{ij}a_{j\acute{i}}e_{\acute{i}}.$$

Jelikož prvky f a e tvoří báze M, potom máme

$$\sum_{i=1}^n a_{ji}b_{i\acute{j}} = \begin{cases} 1 & \text{jestliže } j = \acute{j} \\ 0 & \text{jestliže } j \neq \acute{j} \end{cases}$$

$$\sum_{j=1}^m b_{ij}a_{j\acute{i}} = \begin{cases} 1 & \text{jestliže } i = \acute{i} \\ 0 & \text{jestliže } i \neq \acute{i} \end{cases},$$

kde $j, \acute{j} = 1, 2, ..., m$; $i, \acute{i} = 1, 2, ..., n$. Nyní předpokládejme, že $m < n$ a uvažujme dvě matice typu $n \times n$

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ \dots & \dots & \dots & \dots \\ a_{m1} & a_{m2} & \dots & a_{mn} \\ 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 0 \end{pmatrix}$$

$$B = \begin{pmatrix} b_{11} & \dots & b_{1m} & 0 & \dots & 0 \\ b_{21} & \dots & b_{2m} & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \\ b_{m1} & \dots & b_{nm} & 0 & \dots & 0 \end{pmatrix}$$

Ze vztahů výše pak platí, že $BA = 1$, tedy jednotkové matici. R je komutativní, $det(BA) = det(1) = 1$ je invertibilní v R, tedy i matice A, B jsou invertibilní a platí, že $AB = 1$. Více v [5, Theorem 2.1]. Nicméně je zřejmé, že posledních $n-m$ řádků součinu matic AB je rovno nule, tedy $AB \neq 1$. Dostáváme tedy spor s tím, že $m \geq n$. Analogicky bychom také dostali, že $n \geq m$, tedy $n = m$.
[5, podkapitola 3.4]

Definice 15 *Jestliže R je obor integrity a M je R-modul, potom*

$$Tor(M) = \{x \in M \mid rx = 0 \text{ pro } r \neq 0, r \in R\}$$

je torzní podmodul modulu M. Jestliže $Tor(M) = 0$, nazývá se M beztorzní.

Definice 16 *Pro obory integrity R je stupněm R-modulu M maximální počet R-lineárně nezávislých prvků z M.*

Věta 8 *Nechť R je obor hlavních ideálů, M je volný R-modul konečného stupně n a N je podmodul modulu M. Potom platí:*
1) N je volný stupně m, $m \leq n$ a
2) existuje báze $y_1, y_2, ..., y_n$ z M taková, že $a_1y_1, a_2y_2, ..., a_my_m$ je báze N, kde $a_1, a_2, ..., a_m$ jsou nenulové prvky z R, pro něž platí $a_1 \mid a_2 \mid \cdots \mid a_m$.

Důkaz: Věta platí triviálně pro $N = \{0\}$, takže předpokládejme, že $N \neq \{0\}$. Pro každý homomorfismus φ z R-modulu M do R platí, že obraz $\varphi(N)$ z N je podmodul R, t.j., ideál v R. Jelikož R je okruh hlavních ideálů, pak tento ideál je také hlavní, neboli $\varphi(N) = (a_\varphi)$, pro nějaké $a_\varphi \in R$. Nechť $\Sigma = \{(a_\varphi) \mid \varphi \in Hom_R(M, R)\}$ je množina hlavních ideálů v R získaná z jednotlivých homomorfismů z M do R. Množina Σ je jistě neprázdná, neboť například pro triviální homomorfismus platí, že $(0) \in \Sigma$. Dle Důsledku 3 Σ má nejméně jeden maximální prvek, t.j., existuje alespoň jeden homomorfismus ν z M do R, tedy hlavní ideál $\nu(N) = (a_v)$ není zcela obsažen v jakémkoli jiném prvku množiny Σ. Nechť $a_1 = a_v$ pro tento maximální prvek a nechť $y \in N$ je prvek zobrazující se na generátor a_1 dle homomorfismu $\nu : \nu(y) = a_1$.
Nyní ukážeme, že prvek a_1 není nulový. Nechť $x_1, x_2, ..., x_n$ je báze volného modulu M a nechť $\pi_i \in Hom_R(M, R)$ homomorfismus i-tého prvku vzhledem k této bázi. Jelikož $N \neq \{0\}$, potom existuje i, pro které $\pi_i(N) \neq 0$, tedy Σ obsahuje více než jen triviální ideál (0). Jelikož (a_1) je maximální prvek množiny Σ, pak z toho plyne, že $a_1 \neq 0$.
Dále ukážeme, že prvek a_1 dělí $\varphi(y)$ pro každé $\varphi \in Hom_R(M, R)$. Nechť d je generátor hlavního ideálu generovaného prvkem a_1 a $\varphi(y)$. Potom d je dělitelem a_1 a $\varphi(y)$ v R a $d = r_1a_1 + r_2\varphi(y)$ pro nějaké r_1, $r_2 \in R$. Uvažme homomorfismus $\psi(y) = (r_1\nu + r_2\varphi)(y) = r_1a_1 + r_2\varphi(y) = d$, tedy $d \in \psi(N)$, tedy také $(d) \subseteq \psi(N)$. Ale d_1 je dělitel a_1, tedy také $(a_1) \subseteq (d)$. Potom $(a_1) \subseteq (d) \subseteq \psi(N)$ a dle maximality (a_1) dostaneme: $(a_1) = (d) = \psi(N)$. Jelikož d dělí $\varphi(y)$, také $a_1 \mid \varphi(y)$. Jesltiže toto aplikujeme na homomorfismus π_i, pak je vidět, že a_1 dělí $\pi_i(y)$ pro všechna i. Napišme $\pi_i(y) = a_1b_1$ pro nějaké $b_i \in R$, $1 \leq i \leq n$ a definujme $y_1 = \sum_{i=1}^{n} b_ix_i$. Označme $a_1y_1 = y$. Jelikož $a_1 = \nu(y) = \nu(a_1y_1) = a_1\nu(y_1)$ a a_1 je nenulový prvek oboru integrity R, potom $\nu(y_1) = 1$. Nyní ověříme, že prvek y_1

můžeme použít jako prvek báze M a a_1y_1 jako jeden prvek báze N, tedy

a) $M = Ry_1 \oplus ker\nu$ a

b) $N = Ra_1y_1 \oplus (N \cap ker\nu)$.

a) Nechť x je libovolný prvek v M a napišme $x = \nu(x)y_1 + (x - \nu(x)y_1)$. Jellikož $\nu(x - \nu(x)y_1) = \nu(x) - \nu(x)\nu(y_1) = \nu(x) - \nu(x) \cdot 1 = 0$, pak tedy $x - \nu(x)y_1$ je prvek jádra ν. Tedy x může být psán jako součet prvku v Ry_1 a prvku jádra zobrazení ν, tedy $M = R_1 + ker\nu$. Předpokládejme, že ry_1 je také prvkem jádra zobrazení ν. Potom $0 = \nu(ry_1) = r\nu(y_1) = r$.

b) Víme, že $\nu(\acute{x})$ je dělitelný prvkem a_1 pro všechna $\acute{x} \in N$, dle definice a_1, jakožto generátoru $\nu(N)$. Jestliže napíšeme $\nu(\acute{x}) = ba_1$, kde $b \in R$, potom dle rozkladu užitém pro tvrzení a) výše je $\acute{x} = \nu(\acute{x})y_1 + (\acute{x} - \nu(\acute{x})y_1) = ba_1y_1 + (\acute{x} - ba_1y_1)$, kde druhý sčítanec je v jádře zobrazení ν a je prvkem N. Z toho plyne, že $N = Ra_1y_1 + (N \cap ker\nu)$.
Nyní dokážeme první část věty indukcí podle stupně m podmodulu N. Jestliže $m = 0$, potom N je torzní modul, tedy $N = 0$, neboť N je podmodul modulu M a ten je volný, a tedy beztorzní. Proto 1) je pro $m = 0$ splněna triviálně. Předpokládejme, že $m > 0$, pak z b) víme, že $N = Ra_1y_1 \oplus (N \cap ker\nu)$, tedy N má stupeň m, Ra_1y_1 má stupeň 1 a z toho, že nad oborem integrity je direktní součet dvou modulů roven modulu, který má stupeň rovný součtu stupňů těchto dvou modulů plyne, že $(N \cap ker\nu)$ je stupně $m - 1$. Tím, že $(N \cap ker\nu)$ je podmodul modulu M stupně $m - 1$, pak dle indukčního předpokladu je $(N \cap ker\nu)$ volný. Opět užitím b), kde se jedná o direktní součet dostaneme, že přidáním a_1y_1 k libovolné bázi $(N \cap ker\nu)$ dostaneme bázi N velikosti m. Tedy N je volný.

Druhou část věty dokážeme indukcí podle n, tedy stupně modulu M. Aplikováním prvního tvzení věty na podmodul $ker\nu$ dostaneme, že tento podmodul je volný a protože součet v a) je direktní, je stupně $n-1$. Dle indukčního předpokladu aplikovaného na $ker\nu$ a jeho podmodul $ker\nu \cap N$, vidíme, že prvky $y_2, y_3, ..., y_n$ tvoří bázi $ker\nu$ a prvky $a_2y_2, a_3y_3, ..., a_my_m$ jsou bází $N \cap ker\nu$, kde $a_2, a_3, ..., a_m \in R$ splňující $a_2 \mid a_3 \mid \cdots \mid a_m$. Jelikož součty v a) i b) jsou direktní, $y_1, y_2, ..., y_n$ je báze M a $a_1y_1, a_2y_2, ..., a_my_m$ je báze N.
Zbývá dokázat, že a_1 dělí a_2. Definujme homomorfismus φ z M do R takto $\varphi(y_1) = \varphi(y_2) = 1$ a $\varphi(y_i) = 0$, pro všechna $i > 2$ prvků báze M. Potom pro tento homomorfismus φ máme $a_1 = \varphi(a_1y_1)$, tedy $a_1 \in \varphi(N)$, tedy také $(a_1) \subseteq \varphi(N)$. Z maximality a_1 v Σ plyne, že $(a_1) = \varphi(N)$. Jelikož $a_2 = \varphi(a_2y_2) \in \varphi(N)$ potom máme $a_2 \in (a_1)$ t.j., $a_1 \mid a_2$.
[2, podkapitola 12.1]

Kapitola 3

Smithův normální tvar

Definice 17 *Nechť R je komutativní okruh. Řekneme, že dvě matice $A, B \in M(m \times n, R)$ jsou ekvivalentní, jestliže existují takové dvě invertibilní matice $P \in M(n, R), Q \in M(m, R)$, že platí $QAP^{-1} = B$.*
[5, podkapitola 3.7]

Věta 9 *Nechť R je obor hlavních ideálů. Potom každá matice $A \in M(m \times n, R)$ je ekvivalentní matici*

$$\begin{pmatrix} d_1 & 0 & \dots & . & . & \dots & 0 \\ 0 & d_2 & \dots & . & . & \dots & 0 \\ \vdots & & \ddots & & & & \vdots \\ 0 & . & \dots & d_r & . & \dots & 0 \\ 0 & . & \dots & . & 0 & \dots & 0 \\ \vdots & & & & & \ddots & \vdots \\ 0 & . & \dots & . & . & \dots & 0 \end{pmatrix}$$

splňující $d_i \mid d_{i+1}$ pro $1 \leq i \leq r-1$.
Tato matice se nazývá Smithův normální tvar matice A.

Důkaz: Nechť T_A značí homomorfismus : $R^n \to R^m$ daný násobením zleva maticí A. Potom $T_A(R^n)$ je podmodul volného modulu stupně m. Dle Věty 8 plyne, že existuje báze $\acute{B} = \{e_1, e_2, ..., e_m\}$ modulu R^m a prvky $d_1, d_2, ..., d_r \in R$, pro které platí, že $\{d_1e_1, d_2e_2, ..., d_re_r\}$ je báze $T_A(R^n)$. Nechť $f_i \in R^n, 1 \leq i \leq r$ splňující $T_A(f_i) = d_ie_i$ pro $1 \leq i \leq r$ a N je podmodul R^n generovaný $\{f_1, f_2, ..., f_r\}$. Potom z tvrzení a) v důkazu Věty 8 plyne, že $R^n = N \oplus ker T_A$. Nechť $\{f_{r+1}, f_{r+2}, ..., f_n\}$ je libovolná báze $ker T_A$. Potom $B = \{f_1, f_2, ..., f_n\}$ je báze R^n. Je snadné nahlédnout, že matice lineární transformace vzhledem k bázím B a $\acute{B}$ modulů R^n a R^m je požadovaného tvaru. Neboli, jestliže P značí matici přechodu báze R^n a Q značí matici přechodu báze R^m, potom QAP^{-1} je požadovaného tvaru.
[7, kapitola 3]

Samotný důkaz věty lze podat i bez užití strukturní věty a to konstruktivním způsobem, který zároveň popisuje algoritmus, jak z požadované matice A získat její Smithův normální tvar. Důkaz spočívá v postupných úpravách matice A, které

budou reprezentovány pomocnými maticemi, které zadefinujeme následnovně:

1) Nechť $b \in R$ a nechť $i \neq j$. Položme $T_{ij}(b) = 1 + be_{ij}$, kde e_{ij} je matice s prvkem 1 na pozici $e_{(i,j)}$, na ostatních pozicích rovna nule. $T_{ij}(b)$ je invertibilní, jelikož $T_{ij}(b)T_{ij}(-b) = (1 + be_{ij})(1 - be_{ij}) = 1$.

2) Nechť u je invertibilní prvek R a položme $D_i(u) = 1 + (u - 1)e_{ii}$, tedy $D_i(u)$ je diagonální matice s i-tým prvkem na diagonále rovnému u a ostatní prvky na diagonále rovny 1. Potom $D_i(u)$ je invertibilní, neboť $D_i(u)^{-1} = D_i(u^{-1})$.

3) Nechť $P_{ij} = 1 - e_{ii} - e_{jj} + e_{ij} + e_{ji}$. Také tato matice je invertibilní, jelikož platí, že $P_{ij}^2 = 1$.

Nechť tedy máme matici $A \in M_{mn}(R)$. Samotné elemetární úpravy na matici A lze pak vyjádřit přenásobením matice A jednou z příslušných matic a to následovně:

I)
a) Vynásobením matice A zleva maticí $T_{ij}(b)$ typu $m \times m$ získáme matici, jejíž i-tý řádek je roven násobku j-tého řádku matice A s prvkem b spolu s přičtením k původnímu i-tému řádku matice A. Ostatní řádky matice A zůstanou zachovány.
b) Vynásobením matice A zprava maticí $T_{ij}(b)$ typu $n \times n$ dostaneme matici, jejíž i-tý sloupec je násobkem i-tého sloupce matice A s prvkem b spolu s přičtením j-tého sloupce matice A. Ostatní sloupce matice A zůstanou zachovány.

II)
a) Vynásobením matice A zleva maticí $D_i(u)$ typu $m \times m$ způsobí vynásobení i-tého řádku matice A prvkem u, ostatní řádky zůstanou zachovány.
b) Vynásobením matice A zprava maticí $D_i(u)$ typu $n \times n$ způsobí vynásobení i-tého sloupce matice A prvkem u, ostatní sloupce zůstanou zachovány.

III)
a) Vynásobením matice A zleva maticí P_{ij} typu $m \times m$ způsobí výměnu i-tého a j-tého řádku v matici A, ostatní řádky zůstanou zachovány.
b) Vynásobením matice A zprava maticí P_{ij} typu $n \times n$ způsobí výměnu i-tého a j-tého sloupce v matici A, ostatní sloupce zůstanou zachovány.

Nyní tedy přistoupíme ke konstruktivnímu důkazu Věty 9. Nejprve ukážeme důkaz pro případ, kdy R je Euklidovský obor s Euklidovskou normou $\delta : R \to N \cup \{0\}$. Nechť $A \neq 0$ a nechť a_{ij} je nenulový prvek z A s minimální $\delta(a_{ij})$. Použitím elementárních operací převedeme daný prvek a_{ij} v matici A na pozici $A_{(1,1)}$. Nechť $k > 1$ a $a_{1k} = a_{11}b_k + b_{1k}$, kde $\delta(b_{1k}) < \delta(a_1)$. Nyní odečteme první sloupec výnasobený b_k od k-tého. Tato elementární operace nahradí a_{1k} za b_{1k}. Jestliže $b_{1k} \neq 0$, potom jsme získali matici ekvivalentní matici A, pro kterou minimum normy δ u nenulových prvků je menší než u původní matice A. Původní postup opakujeme pro tuto novou matici. Podobně, pokud $a_{k1} = a_{11}b_k + b_{k1}$, kde $b_{k1} \neq 0$ a $\delta(b_{k1}) < \delta(a_{11})$, potom elementárními úpravami typu I) na řádky matice dostaneme ekvivalentní matici, pro kterou opět bylo sníženo minimum δ u

nenuloých prvků. Jelikož obraz funkce δ pro nenulové prvky je nezáporné kladné konečné číslo, pak aplikováním zmíněného postupu dostaneme ekvivalentní matici $B = (b_{ij})$, pro kterou platí, že $b_{11} \mid b_{1k}$ a $b_{11} \mid b_{k1}$ pro všechna k. Dalšími elementárními úpravami typu I) dostaneme ekvivalentní matici tvaru

$$\acute{A} := \begin{pmatrix} b_{11} & 0 & \dots & 0 \\ 0 & c_{22} & \dots & c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & c_{m2} & \dots & c_{mn} \end{pmatrix}$$

Chceme, aby platilo, že $b_{11} \mid c_{kl}$ pro všechna k, l. Jestliže tedy existuje $c_{kl} : b_{11}$ nedělí c_{kl}, potom přičteme k-tý řádek k prvnímu řádku a dostaneme tak nový řádek $(b_{11}, c_{k2}, \dots, c_{kl}, \dots, c_{kn})$. Opakováním prvního postupu nahradíme c_{kl} nenulovým prvkem s menší normou δ než u prvku b_{11}. Konečným počtem takovýchto kroků dostaneme matici tvaru $\acute{A}$, která je ekvivalentní matici A a a splňuje $b_{11} \neq 0$ a $b_{11} \mid c_{kl}$ pro všechna k, l. Nyní opakujeme proces na podmatici (c_{kl}). Tím dostaneme ekvivalentní matici tvaru

$$\begin{pmatrix} b_{11} & 0 & \cdot & \dots & 0 \\ 0 & c_{22} & 0 & \dots & 0 \\ 0 & 0 & f_{33} & \dots & f_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & f_{m3} & \dots & f_{mn} \end{pmatrix}$$

pro kterou platí, že $c_{22} \mid f_{pq}$ pro všechna p, q. Navíc elementární úpravy řádků a sloupců aplikovaných na podmatici (c_{kl}) neovlivní podmínku dělitelnosti pro prvek b_{11}. Tedy $b_{11} \mid c_{22}$ a tedy $b_{11} \mid f_{pq}$. Rekurzí nakonec dostaneme diagonální matici $\{d_1, d_2, ..., d_r, 0, ..., 0\}$, kde $d_i \mid d_j$ pro $i \leq j, (d_1 = b_{11}, d_2 = c_{22}, ...)$.

V obecném případě, kdy R je okruhem hlavních ideálů, nahradíme funkci δ funkcí $l : R \to N \cup \{0\}$. Libovolný nenulový prvek $a \in R$ se dá jednoznačně vyjádřit jako součin prvočinitelů. Zadefinujme délku $l(a)$ pro $a \neq 0$ jako počet prvočinitelů ve faktorizaci prvku a. Přeněji, pokud $a = p_1 p_2 \dots p_r$, kde p_i nejsou nutně různá, pak $l(a) = r$. Pokud a je jednotka, pak $l(a) = 0$. V úpravách matice A budeme také využívat násobení maticemi tvaru

$$\acute{C} := \begin{pmatrix} x & s & & & & \\ y & t & & & 0 & \\ & & 1 & & & \\ & & & 1 & & \\ & 0 & & & \ddots & \\ & & & & & 1 \end{pmatrix}$$

kde $\begin{pmatrix} x & s \\ y & t \end{pmatrix}$ je invertibilní. Předpokládejme, že $a_{11} \neq 0$ a $l(a_{11}) \leq l(a_{ij})$ pro všechna $a_{ij} \neq 0$. Nechť nastane případ, že existuje a_{1k}, pro které platí, že a_{11} nedělí a_{1k}. Prohozením druhého a k-tého sloupce tedy dostaneme, že a_{11} nedělí

a_{12}. Napišme $a = a_{11}, b = a_{12}$ a nechť $d = NSD(a, b)$, tedy $l(d) < l(a)$. Potom existují prvky $x, y \in R$, pro které platí, že $ax + by = d$. Položme $s = bd^{-1}$, $t = -ad^{-1}$. Potom dostáváme vztah

$$\begin{pmatrix} -t & s \\ y & -x \end{pmatrix} \begin{pmatrix} x & s \\ y & t \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

který implikuje, že obě matice jsou invertibilní (více v [5, Theorem 2.1]). Potom tedy i matice $\acute{C}$ je invertibilní. Vynásobením matice A zprava maticí $\acute{C}$ nám dá matici, jejíž první řádek je $(d, 0, b_{13}, \ldots, b_{1n})$ a $l(d) < l(a_{11})$. Podobně, jestliže a_{11} nedělí a_{k1} pro nějaké k, pak pomocí elementárních úprav spolu s přenásobením matice A zleva maticí $\acute{C}$ dostaneme ekvivalentní matici, ve které délka libovolného nenulového prvku je menší než $l(a_{11})$. Opakováním tohoto postupu dojdeme do situace, kdy $a_{11} \mid a_{1k}$ a $a_{11} \mid a_{k1}$ pro všechna k. Elementárními úpravami převedeme matici na tvar matice $\acute{A}$. Další postup je analogický postupu v případě Euklidovských oborů s využitím funkce l namísto δ.
[5, podkapitola 3.7]

Z předchozího důkazu pro okruh hlavních ideálů pak také plyne následující Tvrzení 2 pro okruhy Z_p^k.

Tvrzení 2 *Nechť máme danou matici $A \in (Z_{p^l})^{m\times n}$. Potom existují matice $L \in (Z_{p^l})^{m\times m}$ a $R \in (Z_{p^l})^{n\times n}$, které jsou invertibilní modulo p^l a dále platí, že LAR je Smithův tvar matice A.*

Důkaz: Pro kladné celé číslo $n = a \cdot p^l$, kde $NSD(a, p) = 1$, budeme l nazývat p-valuací čísla n a značit $\upsilon_p(n) = l$. V matici A najdeme prvek a_{ij}, pro který platí, že $\upsilon_p(a_{ij}) = min\{\upsilon_p(a_{st}) \mid 1 \le s \le m, 1 \le t \le n\}$. Elementárními úpravami matice A přesuneme prvek a_{ij} na pozici $A_{(1,1)}$. Potom $a_{11} = a \cdot p^{\upsilon_p(a_{11})}$ pro nějaké celé číslo a splňující $NSD(a, p) = 1$. Prvek a je invertibilní modulo p^l. Přenásobením prvního řádku inverzním prvkem α k prvku a dostaneme na pozici $A_{(1,1)}$ prvek $\alpha a_{11} \equiv p^{\upsilon_p(a_{11})} \pmod{p^l}$, který dělí všechny prvky a_{i1} pro $2 \le i \le m$ a a_{1j} pro $2 \le j \le n$. Tedy nalezneme matice L_1, R_1, pro které platí, že

$$L_1 A R_1 = \begin{pmatrix} p^{\upsilon_p(a_{11})} & 0 & \ldots & 0 \\ 0 & & & \\ \vdots & & \widetilde{A} & \\ 0 & & & \end{pmatrix}.$$

Pokračováním rekurzivně na podmatici $\widetilde{A}$ nakonec dostaneme matice L, R, pro které platí, že $LAR = D$, kde D je Smithův tvar matice A.
[3]

Kapitola 4

Řešení soustavy rovnic nad okruhem Z

4.1 Popis algoritmu

Nechť máme zadanou soustavu rovnic nad Z tvaru $Ax = b$.

1) Pomocí elementárních úprav chceme matici A převést do Smithova normálního tvaru. Najdeme tedy prvek matice A, který je nenulový a má nejmeší hodnotu, označme tento prvek a. Okruh celých čísel Z je Euklidovský obor, tudíž pro určování hodnot prvků matice A využijeme absolutní hodnotu prvku, která je nad Z Euklidovskou normou. Následně a přesuneme na pozici $A_{(1,1)}$ a chceme, aby dělil ostatní prvky v prvním řádku a prvky v prvním sloupci. Pokud tedy existuje například prvek b na pozici $A_{(1,2)}$, který není dělitelný a, pak z Věty 1 víme, že existují $x, y \in Z : xa + yb = NSD(a, b) = d$. Vynásobením prvního sloupce pvkem x a přičtení y-násobku druhého sloupce k prvnímu sloupci dostaneme na pozici $A_{(1,1)}$ prvek d. Analogicky pokračujeme, dokud prvek na pozici $A_{(1,1)}$ nedělí všechny prvky v prvním řádku a prvky v prvním sloupci. Poté vhodným přenásobením matice A vynulujeme první řádek a první sloupec vyjma pozice $A_{(1,1)}$ a ověříme, zda $a_{1,1}$ dělí všechny zbývající prvky v matici A. Pokud existuje prvek $a_{k,l}$, který není děllitelný prvkem $a_{1,1}$, pak přesuneme k-tý řádek na první řádek a zopakujeme postup uvedený výše, v opačném případě postupujeme analogicky na podmatici počínaje prvkem $a_{2,2}$. Tímto nakonec dostaneme diagonální matici D, která je Smithovým tvarem matice A.

2) Veškeré elementární operace na matici A byly prováděny přenásobením matice A zprava maticemi R_i anebo zleva maticemi L_j. Nechť počet R_i je roven n a počet L_j roven m, potom $R = R_1R_2...R_n$, $L = L_mL_{m-1}...L_1$ a $D = LAR$.

3) Abychom získali řešení soustavy $Ax = b$, spočteme $c = Lb$, tím dostaneme novou pravou stranu a vyřešíme jednoduchou diagonální soustavu $Dy = c$. Hledané řešení je pak $x = Ry$.

4.2 Demonstrace algoritmu na příkladu

Nechť máme zadanou soustavu rovnic nad Z tvaru $Ax = b$, kde

$$A = \begin{pmatrix} 2 & 1 & -3 \\ 7 & 10 & 8 \end{pmatrix}, b = \begin{pmatrix} 13 \\ 26 \end{pmatrix}.$$

1) Najdeme prvek s nejmeší absolutní hodnotou, v tomto případě je to prvek $a_{12} = 1$ a přesuneme ho na pozici $A_{(1,1)}$, t.j. prohodíme první a druhý sloupec. To provedeme vynásobením matice A zprava maticí $P_{12} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} := R_1$.
Výsledkem je matice $A_1 = \begin{pmatrix} 1 & 2 & -3 \\ 10 & 7 & 8 \end{pmatrix}$.

2) $a_{11} \mid a_{1k}$ a $a_{11} \mid a_{k1}$ pro $k > 1$, t.j. vynulujeme první řádek a první sloupec vyjma pozice $A_{(1,1)}$. Tedy matici A_1 postupně přenásobíme zprava maticemi $T_{12}(-2) = \begin{pmatrix} 1 & -2 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} := R_2$, $T_{13}(3) = \begin{pmatrix} 1 & 0 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} := R_3$ a zleva maticí $T_{21}(-10) = \begin{pmatrix} 1 & 0 \\ -10 & 1 \end{pmatrix} := L_1$. Dostaneme matici $A_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -13 & 38 \end{pmatrix}$.

3) Prvek $a_{11} = 1$ dělí všechny nenulové prvky v matici A_2, tedy postup aplikujeme na podmatici počínaje prvkem na pozici $A_{(2,2)}$.

4) Prvek $a_{22} = -13$ upravíme na kladnou hodnotu, tedy matici A_2 přenásobíme zprava maticí $D_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} := R_4$.
Výsledkem je matice $A_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 13 & 38 \end{pmatrix}$.

5) Prvek $a_{22} = 13$ má menší absolutní hodnotu než prvek $a_{23} = 38$. Jelikož 13 nedělí prvek 38, spočteme $NSD(13, 38) = 1$ a určíme koeficienty $x, y : x13 + y38 = 1$. Dostaneme $x = 3, y = -1$, potom tedy vynásobíme druhý sloupec prvkem $x = 3$ a přičteme k němu $y = -1$-násobek třetího sloupce. Tyto operace reprezentujeme pomocí přenásobení matice A_3 zprava maticí $D_2(3) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{pmatrix} = R_5$ a následně přenásobením výsledné matice zprava maticí $T_{32}(-1) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{pmatrix} := R_6$. Výsledkem je matice $A_4 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 38 \end{pmatrix}$.

6) Prvek $a_{22} = 1$ dělí prvek $a_{23} = 38$, tedy vynulujeme pozici $A_{(23)}$ a to přenásobením matice A_5 zprava maticí $T_{23}(-12) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -38 \\ 0 & 0 & 1 \end{pmatrix} := R_7$. Výsledkem je ma-

tice $A_6 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$.

7) Platí, že $a_{11} = 1$ dělí všechny nenulové prvky v A_6 a zároveň je matice A_6 diagonální, t.j., $A_6 = D$ je hledaný Smithův tvar matice A.

8) Jednotlivé operace na matici A jsou zaznamenány v maticích R_i pro $i = 1 \dots 7$ a matici L_1. Nechť tedy $R = R_1R_2R_3R_4R_5R_6R_7 = \begin{pmatrix} 0 & -3 & 114 \\ 1 & -3 & -111 \\ 0 & 1 & 39 \end{pmatrix}$ a $L = L_1 = \begin{pmatrix} 1 & 0 \\ -10 & 1 \end{pmatrix}$. Potom $LAR = D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$.

10) Nyní vyjádříme $c = Lb = \begin{pmatrix} 13 \\ -104 \end{pmatrix}$ a vyřešíme diagonální soustavu rovnic tvaru $Dy = c$, odkud získáme matici $y = \begin{pmatrix} 13 \\ -104 \\ t \end{pmatrix}$, kde $t \in Z$. Hledaným řešením je pak $x = Ry = \begin{pmatrix} 312 + 38t \\ -299 - 37t \\ 104 + 13t \end{pmatrix}$.

4.3 Implementace algoritmu v programu Mathematica

```
(*funkce Tm, Pm, Dm provadeji elementarni operace*)
Tm[matice_, index1_, index2_, strana_, radku_, sloupcu_, prvek_] :=
  Module[{mat, Tmat, i, j, str, rad, sloup, b},
   mat = matice; i = index1; j = index2; str = strana; rad = radku;
   sloup = sloupcu; b = prvek;
   If[str == 1, Tmat = IdentityMatrix[rad];
    Tmat[[i, j]] = (Tmat[[i, j]] + b); mat = Tmat.mat; L = Tmat.L,
    Tmat = IdentityMatrix[sloup]; Tmat[[i, j]] = (Tmat[[i, j]] + b);
    mat = mat.Tmat; R = R.Tmat];
   mat
   ];

Pm[matice_, index1_, index2_, strana_, radku_, sloupcu_] :=
  Module[{mat, Pmat, i, j, str, rad, sloup},
   mat = matice; i = index1; j = index2; str = strana; rad = radku;
   sloup = sloupcu;
   If[str == 1, Pmat = IdentityMatrix[rad];
    Pmat[[i, i]] = (Pmat[[i, i]] - 1);
    Pmat[[j, j]] = (Pmat[[j, j]] - 1);
    Pmat[[i, j]] = (Pmat[[i, j]] + 1);
    Pmat[[j, i]] = (Pmat[[j, i]] + 1); mat = Pmat.mat; L = Pmat.L,
    Pmat = IdentityMatrix[sloup]; Pmat[[i, i]] = (Pmat[[i, i]] - 1);
    Pmat[[j, j]] = (Pmat[[j, j]] - 1);
    Pmat[[i, j]] = (Pmat[[i, j]] + 1);
    Pmat[[j, i]] = (Pmat[[j, i]] + 1); mat = mat.Pmat; R = R.Pmat];
   mat
   ];
```

```
Dm[matice_, index_, strana_, radku_, sloupcu_, prvek_] :=
  Module[{mat, Dmat, i, str, rad, sloup, b},
   mat = matice; i = index; str = strana; rad = radku;
   sloup = sloupcu; b = prvek;
   If[str == 1, Dmat = IdentityMatrix[rad];
    Dmat[[i, i]] = (Dmat[[i, i]] + (b - 1)); mat = Dmat.mat;
    L = Dmat.L, Dmat = IdentityMatrix[sloup];
    Dmat[[i, i]] = (Dmat[[i, i]] + (b - 1)); mat = mat.Dmat;
    R = R.Dmat];
   mat
   ];

(*kontoluje spravny format vstupnich matic*)
Test[matice1_, matice2_] := Module[{mat1, mat2},
   mat1 = matice1; mat2 = matice2;
   If[MatrixQ[A] == False || MatrixQ[B == False], e = False;
    Print["Bad matrix form"],
    If[Length[mat1] != Length[mat2], e = False;
     Print["Bad matrix shape"], e = True]];
   e
   ];

(*najde ve vstupni matici prvek s nejmensi absolutni hodnotou \
a da ho na pozici (i,j)*)
Nej[matice_, index1_, index2_, radku_, sloupcu_] :=
  Module[{mat, i, j, rad, sloup, a, pozR, pozS, m, n},
   mat = matice; i = index1; j = index2; rad = radku;
   sloup = sloupcu; pozR = i; pozS = j; nalez = False;
   m = i; n = j;
   If[mat[[i, j]] == 0,
    While[nalez == False && (m <= rad && n <= sloup),
     If[mat[[m, n]] != 0, a = Abs[mat[[m, n]]]; pozR = m; pozS = n;
      nalez = True]; If[n == sloup, n = j; m++, n++]],
    a = Abs[mat[[i, j]]]; nalez = True];
   If[nalez == True,
    For[k = i, k <= rad, k++,
     For[l = j, l <= sloup, l++,
      If[(Abs[mat[[k, l]]] < a) && (mat[[k, l]] != 0),
       a = Abs[mat[[k, l]]]; pozR = k ; pozS = l]]]];
   If[nalez == True && (i != pozR || j != pozS),
    mat = Pm[mat, i, pozR, 1, rad, sloup];
    mat = Pm[mat, j, pozS, 0, rad, sloup];
    If[mat[[i, j]] < 0, mat = Dm[mat, i, 1, rad, sloup, (-1)]]];
   mat
   ];

(*vynuluje i-ty radek a j-ty sloupec vyjma pozice (i,j)*)
Nul[matice_, index1_, index2_, radku_, sloupcu_] :=
  Module[{mat, i, j, rad, sloup, d},
   mat = matice; rad = radku; sloup = sloupcu; i = index1;
   j := index2; d = 1;
   For[k = (j + 1), k <= sloup, k++,
    If[mat[[i, k]] != 0, d = (mat[[i, k]]/mat[[i, j]]);
     mat = Tm[mat, i, k, 0, rad, sloup, (-d)]]];
   For[l = (i + 1), l <= rad, l++,
    If[mat[[l, j]] != 0, d = (mat[[l, j]]/mat[[i, j]]);
     mat = Tm[mat, l, j, 1, rad, sloup, (-d)]]];
   mat    ];
```

```
(*kontroluje, zda prvek na pozici (i,j) deli vsechny nenulove \
prvky v matici, pokud nejaky nedeli, posune tento radek na \
prvni radek a vyda hodnotu false, aby program vedel, ze \
ma znovu zavolat funkci Nsdb*)
Dele[matice_, index1_, index2_, radku_, sloupcu_] :=
  Module[{mat, i, j, rad, sloup, vysl, k, l, d, poz},
   mat = matice; i = index1; j = index2; rad = radku;
   sloup = sloupcu; vysl = True; k = j + 1; l = i + 1;
   d = 0; poz = 0;
   While[(vysl == True) && (k <= sloup) && (l <= rad),
    d = Mod[mat[[l, k]], mat[[i, j]]];
    If[d != 0, vysl = False; poz = l];
    If[k == sloup, k = (j + 1); l++, k++]];
   If[vysl == False, mat = Tm[mat, i, poz, 1, rad, sloup, 1]];
   {vysl, mat}
   ];

(*upravuje matici elem. upravami dokud neplati, ze prvek na pozici \
(i,j) deli ostatni nenulove prvky v i-tem radku a j-tem sloupci*)
Nsdb[matice_, index1_, index2_, radku_, sloupcu_] :=
  Module[{mat, i, j, rad, sloup, d, nsd, Nsd, x, y, pozR, pozS, vysl1,
    vysl2},
   mat = matice; i = index1; j = index2; rad = radku; sloup = sloupcu;
    pozR = i; pozS = j; vysl1 = False; vysl2 = False;
   While[(vysl1 == False) || (vysl2 == False), vysl1 = True;
    nsd = Abs[mat[[i, j]]]; Nsd = nsd;
    For[k = (j + 1), k <= sloup, k++,
     If[(mat[[i, k]] != 0) && (Mod[mat[[i, k]], mat[[i, j]]] != 0),
      nsd = GCD[mat[[i, k]], mat[[i, j]]];
      If[nsd < Nsd, Nsd = nsd; pozR = i; pozS = k]; vysl1 = False]];
    If[vysl1 == False, {Nsd, {x, y}} =
      ExtendedGCD[mat[[i, j]], mat[[pozR, pozS]]];
     mat = Dm[mat, j, 0, rad, sloup, x];
     mat = Tm[mat, pozS, j, 0, rad, sloup, y]];
    vysl2 = True; nsd = Abs[mat[[i, j]]]; Nsd = nsd;
    For[l = (i + 1), l <= rad, l++,
     If[(mat[[l, j]] != 0) && (Mod[mat[[l, j]], mat[[i, j]]] != 0),
      nsd = GCD[mat[[l, j]], mat[[i, j]]];
      If[nsd < Nsd, Nsd = nsd; pozR = l; pozS = j]; vysl2 = False]];
    If[vysl2 == False, {Nsd, {x, y}} =
      ExtendedGCD[mat[[i, j]], mat[[pozR, pozS]]];
     mat = Dm[mat, i, 1, rad, sloup, x];
     mat = Tm[mat, i, pozR, 1, rad, sloup, y]]];
   mat
   ];

(*najde Smithuv tvar vstupni matice*)
SNF[matice_, radku_, sloupcu_] :=
  Module[{mat, rad, sloup, i, j, vysl},
   mat = matice; rad = radku; sloup = sloupcu; i = 1; j = 1;
   nalez = True;
   While[i <= rad && j <= sloup && nalez == True,
    mat = Nej[mat, i, j, rad, sloup]; vysl = False;
    If[nalez == True,
     While[vysl == False, mat = Nsdb[mat, i, j, rad, sloup];
      mat = Nul[mat, i, j, rad, sloup]; {vysl, mat} =
       Dele[mat, i, j, rad, sloup]];
```

```
      i++; j++]];
    mat
    ];

(*resi soustavu rovnic nad Z*)
SolveZ[matice1_, matice2_] :=
  Module[{mat1, mat2, c, y, rad, sloup, diag, solve, j, i, zbytek,
    hodnostD, h, ch, z, r},
    mat1 = matice1; mat2 = matice2; rad = Length[mat1];
    sloup = Length[mat1[[1]]]; R = IdentityMatrix[sloup];
    L = IdentityMatrix[rad]; e = Test[mat1, mat2]; z = 0; r = 1;
    y = Table[0, {sloup}, {1}];
    If[e == True, diag = SNF[mat1, rad, sloup];
     c = L.mat2; hodnostD = MatrixRank[diag];
     For[a = hodnostD + 1, a <= rad, a++,
      If[c[[a, 1]] != 0, e = False]]];
    h = 1; ch = 1;
    While[h <= hodnostD && e == True,
     z = Mod[c[[h, 1]], diag[[h, ch]]];
     r = Quotient[c[[h, 1]], diag[[h, ch]]];
     If[z == 0, y[[h, 1]] = r, e = False]; h++; ch++];
    If[e == True, j = 1; i = 1;
     While[i <= rad && j <= sloup,
      If[diag[[i, j]] == 0, y[[i, 1]] = Subscript[t, i]]; j++; i++];
     If[sloup > rad, zbytek = sloup - rad;
      For[w = 1, w <= zbytek, w++,
       y[[rad + w, 1]] = Subscript[t, rad + w]]]; solve = R.y; solve,
     Print["No solution over Z"]]
    ];
```

Ukázka programu na příkladu nad Z:

Nechť máme soustavu lineárních rovnic tvaru $Ax = b$, kde x je vektor řešení dané soustavy, potom:

Vstup: $A = \{\{2, 1, -3\}, \{7, 10, 8\}\}; b = \{\{13\}, \{26\}\};$

Zavolání programu: SolveZ[A, b]

Výstup: $\{\{312 + 114t_3\}, \{-299 - 111t_3\}, \{104 + 39t_3\}\}$

Výstupem programu je sloupcový vektor $\begin{pmatrix} x_1 = \{312 + 114t_3\} \\ x_2 = \{-299 - 111t_3\} \\ x_3 = \{104 + 39t_3\} \end{pmatrix}$ odpovídající řešení dané soustavy, kde $t_3 \in Z$ je parametr odlišený indexem pro případ, kdy by bylo parametrů v řešení více.

Kapitola 5

Řešení soustavy rovnic nad okruhem $Q[x]$

5.1 Popis algoritmu

Nechť máme zadanou soustavu rovnic nad $Q[x]$ tvaru $Ax = b$.

1) Pomocí elementárních úprav chceme matici A převést do Smithova normálního tvaru. Najdeme tedy prvek matice A, který je nenulový a má nejmeší hodnotu, označme tento prvek a. Okruh polynomů nad racionálními čísly $Q[x]$ je Euklidovský obor, tudíž pro určování hodnot prvků matice A využijeme stupeň prvku + 1, který je nad $Q[x]$ Euklidovskou normou (program Mathematica přiřazuje nulovému prvku stupeň $-\infty$, proto v programové implementaci stačí uvažovat stupeň daného prvku). Následně a přesuneme na pozici $A_{(1,1)}$ a chceme, aby dělil ostatní prvky v prvním řádku a prvky v prvním sloupci. Pokud tedy existuje například prvek b na pozici $A_{(1,2)}$, který není dělitelný a, pak z Věty 1 víme, že existují $x, y \in Q[x] : xa + yb = NSD(a, b) = d$. Vynásobením prvního sloupce pvkem x a přičtení y-násobku druhého sloupce k prvnímu sloupci dostaneme na pozici $A_{(1,1)}$ prvek d. Analogicky pokračujeme, dokud prvek na pozici $A_{(1,1)}$ nedělí všechny prvky v prvním řádku a prvky v prvním sloupci. Poté vhodným přenásobením matice A vynulujeme první řádek a první sloupce vyjma pozice $A_{(1,1)}$ a ověříme, zda $a_{1,1}$ dělí všechny zbývající prvky v matici A. Pokud existuje prvek $a_{k,l}$, který není děllitelný prvkem $a_{1,1}$, pak přesuneme k-tý řádek na první řádek a zopakujeme postup uvedený výše, v opačném případě postupujeme analogicky na podmatici počínaje prvkem $a_{2,2}$. Tímto nakonec dostaneme diagonální matici D, která je Smithovým tvarem matice A.

2) Veškeré elementární operace na matici A byly prováděny přenásobením matice A zprava maticemi R_i anebo zleva maticemi L_j. Nechť počet R_i je roven n a počet L_j roven m, potom $R = R_1R_2...R_n$, $L = L_mL_{m-1}...L_1$ a $D = LAR$.

3) Abychom získali řešení soustavy $Ax = b$, spočteme $c = Lb$, tím dostaneme novou pravou stranu a vyřešíme jednoduchou diagonální soustavu $Dy = c$. Hledané řešení je pak $x = Ry$.

5.2 Demonstrace algoritmu na příkladu

Nechť máme zadanou soustavu rovnic nad $Q[x]$ tvaru $Ax = b$, kde

$$A = \begin{pmatrix} 1+x^2 & x \\ 1+x & 1+x^3 \end{pmatrix}, b = \begin{pmatrix} 1-x+x^3+x^5 \\ 0 \end{pmatrix}.$$

1) Najdeme nenulový prvek nejnižší normy, vezmeme tedy prvek $a_{12} = x$, pro něhož platí $deg(x) = 1$. Prvek a_{12} přesuneme na pozici $A_{(11)}$ a to vynásobením matice A zprava maticí $P_{12} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} := R_1$. Dostaneme tak výslednou matici $A_1 = \begin{pmatrix} x & 1+x^2 \\ 1+x^3 & 1+x \end{pmatrix}$.

2) Prvek $a_{11} = x$ nedělí prvek $a_{13} = 1 + x^3$, určíme tedy $NSD(x, 1 + x^3) = 1$ a najdeme koeficienty $a, b : ax + b(1 + x^3) = 1$, kde $a, b \in Q[x]$. Nechť tedy $a = -x^2, b = 1$, potom přenásobíme první řádek prvkem $-x^2$ a přičteme k němu 1-násobek druhého řádku. Tyto operace provedeme vynásobením matice A_1 zleva maticí $D_1(-x^2) = \begin{pmatrix} -x^2 & 0 \\ 0 & 1 \end{pmatrix} := L_1$ a výslednou matici přenásobíme zleva maticí $T_{12}(1) = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} := L_2$. Dostaneme tak matici
$A_2 = \begin{pmatrix} 1 & 1+x-x^2-x^4 \\ 1+x^3 & 1+x \end{pmatrix}$.

3) Prvek $a_{11} = 1$ dělí prvky $a_{12} = 1 + x - x^2 - x^4$, $a_{21} = 1 + x^3$, vynulujme tedy první řádek a první sloupec vyjma pozice $A_{2_{(1,1)}}$ a to vynásobením matice A_2 zprava maticí $T_{12} = \begin{pmatrix} 1 & -1-x+x^2+x^4 \\ 0 & 1 \end{pmatrix} := R_2$ a následně vynásobením výsledné matice zleva maticí $T_{21} = \begin{pmatrix} 1 & 0 \\ -1-x^3 & 1 \end{pmatrix} := L_3$. Dostaneme tak matici
$A_3 = \begin{pmatrix} 1 & 0 \\ 0 & x^2-x^3+x^5+x^7 \end{pmatrix}$.

4) Prvek a_{11} dělí všechny nenulové prvky matice A_3 a zároveň je A_3 diagonální, tedy $A_3 = D$ je hledaný Smithův tvar matice A.

5) Nechť $R = R_1R_2 = \begin{pmatrix} 0 & 1 \\ 1 & -1-x+x^2+x^4 \end{pmatrix}$, $L = L_3L_2L_1 =$
$\begin{pmatrix} -x^2 & 1 \\ x^2+x^5 & -x^3 \end{pmatrix}$. Potom platí, že $LAR = D = \begin{pmatrix} 1 & 0 \\ 0 & x^2-x^3+x^5+x^7 \end{pmatrix}$.

6) Nyní vyjádříme $c = Lb = \begin{pmatrix} -x^2+x^3-x^5-x^7 \\ x^2-x^3+2x^5-x^6+x^7+x^8+x^{10} \end{pmatrix}$ a vyřešíme diagonální soustavu rovnic tvaru $Dy = c$, odkud získáme matici
$y = \begin{pmatrix} -x^2+x^3-x^5-x^7 \\ 1+x^3 \end{pmatrix}$. Hledaným řešením je pak $x = Ry = \begin{pmatrix} 1+x^3 \\ -1-x \end{pmatrix}$.

5.3 Implementace algoritmu v programu Mathematica

```
(*funkce Tm, Pm, Dm provadeji elementarni operace*)
Tm[matice_, index1_, index2_, strana_, radku_, sloupcu_, prvek_] :=
  Module[{mat, Tmat, i, j, str, rad, sloup, b},
   mat = matice; i = index1; j = index2; str = strana; rad = radku;
   sloup = sloupcu; b = prvek;
   If[str == 1, Tmat = IdentityMatrix[rad];
    Tmat[[i, j]] = (Tmat[[i, j]] + b); mat = Tmat.mat; L = Tmat.L,
    Tmat = IdentityMatrix[sloup]; Tmat[[i, j]] = (Tmat[[i, j]] + b);
    mat = mat.Tmat; R = R.Tmat];
   mat
   ];

Pm[matice_, index1_, index2_, strana_, radku_, sloupcu_] :=
  Module[{mat, Pmat, i, j, str, rad, sloup},
   mat = matice; i = index1; j = index2; str = strana; rad = radku;
   sloup = sloupcu;
   If[str == 1, Pmat = IdentityMatrix[rad];
    Pmat[[i, i]] = (Pmat[[i, i]] - 1);
    Pmat[[j, j]] = (Pmat[[j, j]] - 1);
    Pmat[[i, j]] = (Pmat[[i, j]] + 1);
    Pmat[[j, i]] = (Pmat[[j, i]] + 1); mat = Pmat.mat; L = Pmat.L,
    Pmat = IdentityMatrix[sloup]; Pmat[[i, i]] = (Pmat[[i, i]] - 1);
    Pmat[[j, j]] = (Pmat[[j, j]] - 1);
    Pmat[[i, j]] = (Pmat[[i, j]] + 1);
    Pmat[[j, i]] = (Pmat[[j, i]] + 1); mat = mat.Pmat; R = R.Pmat];
   mat
   ];

Dm[matice_, index_ , strana_ , radku_, sloupcu_, prvek_] :=
  Module[{mat, Dmat, i, str, rad, sloup, b},
   mat = matice; i = index; str = strana; rad = radku;
   sloup = sloupcu; b = prvek;
   If[str == 1, Dmat = IdentityMatrix[rad];
    Dmat[[i, i]] = (Dmat[[i, i]] + (b - 1)); mat = Dmat.mat;
    L = Dmat.L, Dmat = IdentityMatrix[sloup];
    Dmat[[i, i]] = (Dmat[[i, i]] + (b - 1)); mat = mat.Dmat;
    R = R.Dmat];
   mat
   ];

(*kontoluje spravny format vstupnich matic*)
Test[matice1_, matice2_] := Module[{mat1, mat2},
   mat1 = matice1; mat2 = matice2;
   If[MatrixQ[A] == False || MatrixQ[B == False], e = False;
    Print["Bad matrix form"],
    If[Length[mat1] != Length[mat2], e = False;
     Print["Bad matrix shape"], e = True]];
   e
   ];

(*najde ve vstupni matici prvek s nejmensim stupnem a da ho \
na pozici (i,j)*)
Nej[matice_, index1_, index2_, radku_, sloupcu_, promenna_] :=
  Module[{mat, i, j, rad, sloup, a, pozR, pozS, vysl, m, n, prom},
```

```
    mat = matice; i = index1; j = index2; rad = radku; sloup = sloupcu;
     pozR = i; pozS = j; nalez = False; m = i; n = j; prom = promenna;
    If[Exponent[mat[[i, j]], prom] < 0,
     While[nalez == False && (m <= rad && n <= sloup),
      If[Exponent[mat[[m, n]], prom] >= 0,
       a = Exponent[mat[[m, n]], prom]; pozR = m; pozS = n;
       nalez = True]; If[n == sloup, n = j; m++, n++]],
     a = Exponent[mat[[i, j]], prom]; nalez = True];
    If[nalez == True,
     For[k = i, k <= rad, k++,
      For[l = j, l <= sloup, l++,
       If[(Exponent[mat[[k, l]], prom] <
            a) && (Exponent[mat[[k, l]], prom] >= 0),
        a = Exponent[mat[[k, l]], prom]; pozR = k ; pozS = l]]]];
    If[nalez == True, mat = Pm[mat, i, pozR, 1, rad, sloup];
     mat = Pm[mat, j, pozS, 0, rad, sloup]];
    mat
    ];

(*vynuluje i-ty radek a j-ty sloupec vyjma pozice (i,j)*)
Nul[matice_, index1_, index2_, radku_, sloupcu_, promenna_] :=
  Module[{mat, i, j, rad, sloup, d, prom},
    mat = matice; rad = radku; sloup = sloupcu; i = index1;
    j := index2; d = 1; prom = promenna;
    For[k = (j + 1), k <= sloup, k++,
     If[Exponent[mat[[i, k]], prom] >= 0,
      d = PolynomialQuotient[mat[[i, k]], mat[[i, j]], prom];
      mat = Tm[mat, i, k, 0, rad, sloup, (-d)]]];
    For[l = (i + 1), l <= rad, l++,
     If[Exponent[mat[[l, j]], prom] >= 0,
      d = PolynomialQuotient[mat[[l, j]], mat[[i, j]], prom];
      mat = Tm[mat, l, j, 1, rad, sloup, (-d)]]];
    mat
    ];

(*kontroluje, zda prvek na pozici (i,j) deli vsechny nenulove \
prvky v matici, pokud nejaky nedeli,
posune tento radek na prvni radek a vyda hodnotu false, \
aby program vedel, ze ma znovu zavolat funkci Nsdb*)
Dele[matice_, index1_, index2_, radku_, sloupcu_, promenna_] :=
  Module[{mat, i, j, rad, sloup, vysl, k, l, d, poz, prom},
    mat = matice; i = index1; j = index2; rad = radku;
    sloup = sloupcu; vysl = True; k = j + 1; l = i + 1; d = 0;
    poz = 0; prom = promenna;
    If[Exponent[mat[[i, j]], prom] >= 0,
     While[(vysl == True) && (k <= sloup) && (l <= rad),
      d = PolynomialRemainder[mat[[l, k]], mat[[i, j]], prom];
      If[Exponent[d, prom] >= 0, vysl = False; poz = l];
      If[k == sloup, k = (j + 1); l++, k++]]];
    If[vysl == False, mat = Tm[mat, i, poz, 1, rad, sloup, 1]];
    {vysl, mat}
    ];

(*upravuje matici elem. upravami dokud neplati, ze prvek na pozici \
(i,j) deli ostatni nenulove prvky v i-tem radku a j-tem sloupci*)
Nsdb[matice_, index1_, index2_, radku_, sloupcu_, promenna_] :=
  Module[{mat, i, j, rad, sloup, d, nsd, Nsd, u, v, pozR, pozS, vysl1,
     vysl2, prom},
```

```
    mat = matice; i = index1; j = index2; rad = radku; sloup = sloupcu;
     pozR = i; pozS = j; vysl1 = False; vysl2 = False; prom = promenna;
    If[Exponent[mat[[i, j]], prom] >= 0,
     While[(vysl1 == False) || (vysl2 == False), vysl1 = True;
      nsd = mat[[i, j]]; Nsd = nsd;
      For[k = (j + 1), k <= sloup, k++,
       If[(Exponent[mat[[i, k]], prom] >=
            0) && (Exponent[
             PolynomialRemainder[mat[[i, k]], mat[[i, j]], prom],
             prom] >= 0), {nsd, {u, v}} =
         PolynomialExtendedGCD[mat[[i, k]], mat[[i, j]], prom];
        If[Exponent[nsd, prom] < Exponent[Nsd, prom], Nsd = nsd;
         pozR = i; pozS = k]; vysl1 = False]];
      If[vysl1 == False, {Nsd, {u, v}} =
        PolynomialExtendedGCD[mat[[i, j]], mat[[pozR, pozS]], prom];
       mat = Dm[mat, j, 0, rad, sloup, u];
       mat = Tm[mat, pozS, j, 0, rad, sloup, v]];
      vysl2 = True; nsd = mat[[i, j]]; Nsd = nsd;
      For[l = (i + 1), l <= rad, l++,
       If[(Exponent[mat[[l, j]], prom] >=
            0) && (Exponent[
             PolynomialRemainder[mat[[l, j]], mat[[i, j]], prom],
             prom] >= 0), {nsd, {u, v}} =
         PolynomialExtendedGCD[mat[[l, j]], mat[[i, j]], prom];
        If[Exponent[nsd, prom] < Exponent[Nsd, prom], Nsd = nsd;
         pozR = l; pozS = j]; vysl2 = False]];
      If[vysl2 == False, {Nsd, {u, v}} =
        PolynomialExtendedGCD[mat[[i, j]], mat[[pozR, pozS]], prom];
       mat = Dm[mat, i, 1, rad, sloup, u];
       mat = Tm[mat, i, pozR, 1, rad, sloup, v]]]];
    mat
    ];

(*najde Smithuv tvar vstupni matice*)
SNF[matice_, radku_, sloupcu_, promenna_] :=
  Module[{mat, rad, sloup, i, j, vysl, k, l, jmenovatel, prom},
    mat = matice; rad = radku; sloup = sloupcu; i = 1; j = 1;
    prom = promenna; nalez = True;
    While[i <= rad && j <= sloup && nalez == True,
     mat = Nej[mat, i, j, rad, sloup, prom]; vysl = False;
     If[nalez == True,
      While[vysl == False, mat = Nsdb[mat, i, j, rad, sloup, prom];
       mat = Nul[mat, i, j, rad, sloup, prom]; {vysl, mat} =
        Dele[mat, i, j, rad, sloup, prom]];
      i++; j++]];
    mat
    ];

(*resi soustavu rovnic nad Q[promenna]*)
SolveQ[matice1_, matice2_, promenna_] :=
  Module[{mat1, mat2, c, y, rad, sloup, diag, solve, j, i, zbytek,
    prom, hodnostD, h, ch, z, r},
    mat1 = matice1; mat2 = matice2; rad = Length[mat1];
    sloup = Length[mat1[[1]]]; prom = promenna;
    R = IdentityMatrix[sloup]; L = IdentityMatrix[rad];
    e = Test[mat1, mat2]; z = 0; r = 1; y = Table[0, {sloup}, {1}];
    If[e == True, diag = SNF[mat1, rad, sloup, prom];
     c = L.mat2 ;
```

```
 hodnostD = MatrixRank[diag];
 For[a = hodnostD + 1, a <= rad, a++,
  If[Exponent[c[[a, 1]], prom] >= 0, e = False]]];
h = 1; ch = 1;
While[h <= hodnostD && e == True, z = (c[[h, 1]])/(diag[[h, ch]]);
 y[[h, 1]] = z; h++; ch++];
solve = Simplify[R.y]; h = 1;
While[h <= sloup && e == True,
 If[PolynomialQ[solve[[h, 1]], prom] == False, e = False]; h++];
If[e == True, j = 1; i = 1;
 While[i <= rad && j <= sloup,
  If[diag[[i, j]] == 0, y[[i, 1]] = Subscript[t, i]]; j++; i++];
 If[sloup > rad, zbytek = sloup - rad;
  For[w = 1, w <= zbytek, w++,
   y[[rad + w, 1]] = Subscript[t, rad + w]]]; solve = R.y;
 Simplify[solve], Print["No solution over Q[x]"]]
];
```

Ukázka programu na příkladu nad $Q[x]$:

Nechť máme zadanou soustavu rovnic tvaru $Ax = b$, kde x je sloupcový vektor řešení dané soustavy, potom:

Vstup: $A = \{\{1 + x^2, x\}, \{1 + x, 1 + x^3\}\}; b = \{\{1 - x + x^3 + x^5\}, \{0\}\};$

Zavolání programu: SolveQ[A, b, x]

Výstup: $\{\{1 + x^3\}, \{-1 - x\}\}$

Výstupem programu je sloupcový vektor $\begin{pmatrix} x_1 = \{1 + x^3\} \\ x_2 = \{-1 - x\} \end{pmatrix}$, který odpovídá řešení dané soustavy.

Kapitola 6

Řešení soustavy rovnic nad okruhem Z_m

U okruhu Z_m rozlišíme tři případy podle m.
1) $m = p$, kde p je prvočíslo.Tedy $Z_m = Z_p$ je těleso.
2) $m = p^k$, kde p je prvočíslo a $k > 1$, $k \in N$.
3) $m = p_1^{k_1} \cdot p_2^{k_2} \dots p_n^{k_n}$, kde pro všechna i platí, že p_i jsou prvočísla, $k_i \in N$ a existují i, j, že $p_i \neq p_j$.

6.1 Z_p

6.1.1 Popis algoritmu

Nechť máme zadanou soustavu rovnic nad Z_p, p prvočíslo, tvaru $Ax = b$.

1) Pomocí elementárních úprav chceme matici A převést do Smithova normálního tvaru. Okruh Z_p je těleso, tedy i Euklidovským oborem. Pro určování hodnot prvků matice A použijeme funkci, která každému nenulovému prvku přiřadí prvek jedna, nulovému prvku přiřadí nulu. Takováto funkce je nad tělesem Euklidovskou funkcí. Stačí tedy najít nenulový prvek matice A, označme tento prvek a. Následně a přesuneme na pozici $A_{(1,1)}$ a chceme, aby dělil ostatní prvky v prvním řádku a prvky v prvním sloupci.V tělese ovšem ke každému nenulovému prvku existuje multiplikativní inverz, tedy najdeme k prvku a prvek b, pro který platí $ab \equiv 1 \; mod \; p$. Vynásobením prvního sloupce prvkem b dostaneme na pozici $A_{(1,1)}$ prvek 1. Prvek 1 ovšem dělí všechny prvky, můžeme tedy rovnou pokračovat tím, že vynulujeme první řádek a první sloupce vyjma pozice $A_{(1,1)}$. Poté postupujeme analogicky na podmatici počínaje prvkem $a_{2,2}$. Tímto nakonec dostaneme diagonální matici D, která je Smithovým tvarem matice A.

2) Veškeré elementární operace na matici A byly prováděny přenásobením matice A zprava maticemi R_i anebo zleva maticemi L_j. Nechť počet R_i je roven n a počet L_j roven m, potom $R = R_1 R_2 ... R_n$, $L = L_m L_{m-1} ... L_1$ a $D = LAR$.

3) Abychom získali řešení soustavy $Ax = b$, spočteme $c = Lb$, tím dostaneme novou pravou stranu a vyřešíme jednoduchou diagonální soustavu $Dy = c$. Hledané řešení je pak $x = Ry$.

6.1.2 Demonstrace algoritmu na příkladu

Nechť máme zadanou soustavu rovnic nad Z_p, $p = 5$, tvaru $Ax = b$, kde

$$A = \begin{pmatrix} 0 & 3 & 0 \\ 4 & 1 & 0 \\ 2 & 3 & 2 \end{pmatrix}, b = \begin{pmatrix} 2 \\ 1 \\ 3 \end{pmatrix}.$$

1) Prvek na pozici $A_{(1,1)}$ je roven nule, najdeme tedy první nenulový prvek. Nechť jím je prvek $a_{12} = 3$, potom přesuneme prvek a_{12} na pozici $A_{(1,1)}$ výměnou prvního a druhého sloupce a to přenásobením matice A zprava maticí $P_{12} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} := R_1$. Dostaneme tak matici $A_1 = \begin{pmatrix} 3 & 0 & 0 \\ 1 & 4 & 0 \\ 3 & 2 & 2 \end{pmatrix}$.

2) Z_5 je těleso, najdeme tedy k prvku $a_{11} = 3$ multiplikativní inverz, t.j. prvek b vyhovující rovnici $3b \equiv 1 \bmod 5$. Vynásobíme tedy první řádek prvkem $b = 2$ a dostaneme tak na pozici $A_{1_{(1,1)}}$ prvek $a_{11} = 1$. Přenásobíme tedy matici A_1 zleva maticí $D_1(2) = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} := L_1$. Operace násobení i sčítání jsou prováděny modulo 5. Dostaneme tak matici $A_2 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 4 & 0 \\ 3 & 2 & 2 \end{pmatrix}$.

3) Nyní chceme na pozicích $a_{21} = 1$, $a_{31} = 3$ dostat nuly. Najdeme tedy prvky b, c vyhovující rovnicím $1 + b \equiv 0 \bmod 5$ a $3 + c \equiv 0 \bmod 5$, t.j. $b = 4$ a $c = 2$. Ke druhému řádku přičteme 4-násobek prvního řádku a ke třetímu řádku přičteme 2-násobek prvního řádku. Dané úpravy provedeme přenásobením matice A_2 zleva maticí $T_{21}(4) = \begin{pmatrix} 1 & 0 & 0 \\ 4 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} := L_2$ a následně výslednou matici přenásobíme zleva maticí $T_{31}(2) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 2 & 0 & 1 \end{pmatrix} := L_3$. Výsledná matice je $A_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 2 & 2 \end{pmatrix}$.

4) Analogicky postupujeme na submatici počínaje prvkem $a_{22} = 4$. Najdeme tedy multiplikativní inverz b k prvku $a_{22} = 4$, tj., $b = 4$. Vynásobíme druhý řádek prvkem $b = 4$ a to přenásobením matice A_3 zleva maticí $D_2(4) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 1 \end{pmatrix} := L_4$. Výsledkem je matice $A_4 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 2 \end{pmatrix}$.

5) Nyní chceme vynulovat prvek na pozici $A_{4_{(3,2)}}$. Najdeme tedy aditivní inverz c k prvku $a_{32} = 2$, t.j. $c = 3$ a přenásobíme matici A_4 zleva maticí $T_{32}(3) =$

$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 3 & 1 \end{pmatrix} := L_5$. Dostaneme matici $A_5 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2 \end{pmatrix}$.

6) Multiplikativní inverz b k prvku $a_{33} = 2$ je roven $b = 3$, tedy přenásobíme matici A_5 zleva maticí $D_3(3) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{pmatrix} := L_6$. Výsledkem je matice $A_6 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} = D$, která je hledaným Smithovým tvarem matice A.

7) Nechť tedy $R = R_1 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ a $L = L_6L_5L_4L_3L_2L_1 = \begin{pmatrix} 2 & 0 & 0 \\ 2 & 4 & 0 \\ 0 & 1 & 3 \end{pmatrix}$, pak platí, že $LAR = D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

8) Nyní vyjádříme $c = Lb = \begin{pmatrix} 4 \\ 3 \\ 0 \end{pmatrix}$ a vyřešíme diagonální soustavu rovnic tvaru $Dy = c$, odkud získáme matici $y = \begin{pmatrix} 4 \\ 3 \\ 0 \end{pmatrix}$. Hledaným řešením je pak $x = Ry = \begin{pmatrix} 3 \\ 4 \\ 0 \end{pmatrix}$.

6.1.3 Implementace algoritmu v programu Mathematica

```
(*funkce Tm, Pm, Dm provadeji elementarni operace*)
Tm[matice_, index1_, index2_, strana_, radku_, sloupcu_, prvek_,
    prvocislo_] := Module[{mat, Tmat, i, j, str, rad, sloup, b, prv},
    mat = matice; i = index1; j = index2; str = strana; rad = radku;
    sloup = sloupcu; b = prvek; prv = prvocislo;
    If[str == 1, Tmat = IdentityMatrix[rad];
     Tmat[[i, j]] = (Tmat[[i, j]] + b); mat = Tmat.mat;
     L = Mod[Tmat.L, prv], Tmat = IdentityMatrix[sloup];
     Tmat[[i, j]] = (Tmat[[i, j]] + b); mat = mat.Tmat;
     R = Mod[R.Tmat, prv]];
    Mod[mat, prv]
    ];

Pm[matice_, index1_, index2_, strana_, radku_, sloupcu_, prvocislo_] :=
    Module[{mat, Pmat, i, j, str, rad, sloup, prv},
    mat = matice; i = index1; j = index2; str = strana; rad = radku;
    sloup = sloupcu; prv = prvocislo;
    If[str == 1, Pmat = IdentityMatrix[rad];
     Pmat[[i, i]] = (Pmat[[i, i]] - 1);
     Pmat[[j, j]] = (Pmat[[j, j]] - 1);
     Pmat[[i, j]] = (Pmat[[i, j]] + 1);
     Pmat[[j, i]] = (Pmat[[j, i]] + 1); mat = Pmat.mat;
     L = Mod[Pmat.L, prv], Pmat = IdentityMatrix[sloup];
     Pmat[[i, i]] = (Pmat[[i, i]] - 1);
```

```
    Pmat[[j, j]] = (Pmat[[j, j]] - 1);
    Pmat[[i, j]] = (Pmat[[i, j]] + 1);
    Pmat[[j, i]] = (Pmat[[j, i]] + 1); mat = mat.Pmat;
    R = Mod[R.Pmat, prv]];
   Mod[mat, prv]
   ];

Dm[matice_, index_, strana_, radku_, sloupcu_, prvek_, prvocislo_] :=
    Module[{mat, Dmat, i, str, rad, sloup, b, prv},
   mat = matice; i = index; str = strana; rad = radku;
   sloup = sloupcu; b = prvek; prv = prvocislo;
   If[str == 1, Dmat = IdentityMatrix[rad];
    Dmat[[i, i]] = (Dmat[[i, i]] + (b - 1)); mat = Dmat.mat;
    L = Mod[Dmat.L, prv], Dmat = IdentityMatrix[sloup];
    Dmat[[i, i]] = (Dmat[[i, i]] + (b - 1)); mat = mat.Dmat;
    R = Mod[R.Dmat, prv]];
   Mod[mat, prv]
   ];

(*kontoluje spravny format vstupnich matic a zda se jedna o prvocislo*)
Test[matice1_, matice2_, prvocislo_] := Module[{mat1, mat2, prv},
   mat1 = matice1; mat2 = matice2; prv = prvocislo;
   If[MatrixQ[A] == False || MatrixQ[B == False], e = False;
    Print["Bad matrix form"],
    If[PrimeQ[prv] == False, e = False; Print[prv "is not prime"],
     If[Length[mat1[[1]]] != Length[mat2], e = False;
      Print["Bad matrix shape"], e = True]]];
   e
   ];

(*najde ve vstupni matici nenulovy prvek, da ho na pozici (i,j) a \
alem. operacemi ho prevede na 1 *)
Nej[matice_, index1_, index2_, radku_, sloupcu_, prvocislo_] :=
  Module[{mat, i, j, rad, sloup, a, pozR, pozS, vysl, m, n, prv, cinv},
   mat = matice; i = index1; j = index2; rad = radku; sloup = sloupcu;
    pozR = i; pozS = j; vysl = False; m = i; n = j; prv = prvocislo;
   If[mat[[i, j]] == 0,
    While[vysl == False && (m <= rad && n <= sloup),
     If[mat[[m, n]] != 0, a = mat[[m, n]]; pozR = m; pozS = n;
      vysl = True]; If[n == sloup, n = j; m++, n++]], a = mat[[i, j]];
     vysl = True];
   If[vysl == True, mat = Pm[mat, i, pozR, 1, rad, sloup, prv];
    mat = Pm[mat, j, pozS, 0, rad, sloup, prv]; mat = Mod[mat, prv];
    a = Mod[a, prv];
    If[a != 0, cinv = PowerMod[a, -1, prv];
     mat = Dm[mat, i, 1, rad, sloup, cinv, prv], er = "err"]];
   Mod[mat, prv]
   ];

(*vynuluje i-ty radek a j-ty sloupec vyjma pozice (i,j)*)
Nul[matice_, index1_, index2_, radku_, sloupcu_, prvocislo_] :=
  Module[{mat, i, j, rad, sloup, d, prv},
   mat = matice; rad = radku; sloup = sloupcu; i = index1;
   j := index2; d = 1; prv = prvocislo;
   For[k = (j + 1), k <= sloup, k++,
    If[mat[[i, k]] != 0, d = (mat[[i, k]]/mat[[i, j]]);
     mat = Tm[mat, i, k, 0, rad, sloup, (-d), prv]]];
   For[l = (i + 1), l <= rad, l++,
```

```
    If[mat[[l, j]] != 0, d = (mat[[l, j]]/mat[[i, j]]);
     mat = Tm[mat, l, j, 1, rad, sloup, (-d), prv]]];
   Mod[mat, prv]
   ];

(*najde Smithuv tvar vstupni matice*)
SNF[matice_, radku_, sloupcu_, prvocislo_] :=
  Module[{mat, rad, sloup, i, j, vysl, prv},
   mat = matice; rad = radku; sloup = sloupcu; i = 1; j = 1;
   prv = prvocislo;
   While[i <= rad && j <= sloup && er != "err",
    mat = Nej[mat, i, j, rad, sloup, prv];
    If[er != "err", mat = Nul[mat, i, j, rad, sloup, prv]];
    i++; j++];
   Mod[mat, prv]
   ];

(*resi soustavu rovnic nad nad Z_p*)
SolveZp[matice1_, matice2_, prvocislo_] :=
  Module[{mat1, mat2, c, y, rad, sloup, diag, solve, zbytek, prv, hom,
     hodnostD, h, ch},
   prv = prvocislo; mat1 = matice1; mat2 = matice2;
   R = IdentityMatrix[sloup]; L = IdentityMatrix[rad];
   e = Test[mat1, mat2, prv];
   If[e == True, rad = Length[mat1]; sloup = Length[mat1[[1]]];
    er = "o"; y = Table[0, {sloup}, {1}];
    diag = SNF[mat1, rad, sloup, prv];
    c = L.mat2; c = Mod[c, prv];
    hodnostD = MatrixRank[diag];
    For[a = hodnostD + 1, a <= rad, a++,
     If[c[[a, 1]] != 0, e = False]]];
   If[e == True, h = 1; ch = 1;
    While[h <= hodnostD, y[[h, 1]] = c[[h, 1]]; h++; ch++];
    While[h <= rad && ch <= sloup, y[[h, 1]] = Subscript[t, h]; h++;
     ch++]; If[sloup > rad, zbytek = sloup - rad;
     For[w = 1, w <= zbytek, w++,
      y[[rad + w, 1]] = Subscript[t, rad + w]]];
    solve = Mod[R.y, prv]; solve, Print["No solution over Zp"]]
   ];
```

Ukázka programu na příkladu nad Z_5:

Nechť máme zadanou soustavu lineárních rovnic tvaru $Ax = b$, kde x je sloupcový vektor řešení dané soustavy, potom:

Vstup: $A = \{\{0,3,0\},\{4,1,0\},\{2,3,2\}\}; b = \{\{2\},\{1\},\{3\}\};$

Zavolání programu: SolveZp[A, b, 5]

Výstup: $\{\{3\},\{4\},\{0\}\}$

Výstupem programu je sloupcový vektor $\begin{pmatrix} x_1 = \{3\} \\ x_2 = \{4\} \\ x_3 = \{0\} \end{pmatrix}$, který odpovídá řešení dané soustavy.

6.2 Z_{p^k}

6.2.1 Popis algoritmu

Nechť máme zadanou soustavu rovnic nad Z_{p^k}, p prvočíslo, $k \in N$, tvaru $Ax = b$.

1) Pomocí elementárních úprav chceme matici A převést do Smithova normálního tvaru. Pro určování hodnot prvků matice A zde použijeme funkci, která každému prvku přiřadí p-valuaci. Najdeme tedy prvek matice A, který je nenulový a má nejmeší hodnotu, označme tento prvek a a nechť p-valuace prvku a je rovna l. Následně a přesuneme na pozici $A_{(1,1)}$ a chceme, aby dělil ostatní prvky v prvním řádku a prvky v prvním sloupci. Prvek $a = p^l * c$, tedy $NSD(c, p^k) = 1$ a k prvku c existuje multiplikativní inverz b. Vynásobením prvního sloupce pvkem b dostaneme na pozici $A_{(1,1)}$ prvek p^l. Vzhledem k tomu, že jsme vybrali prvek s nejmenší p-valuací, pak p^l již dělí ostatní prvky v matici A. Můžeme tedy rovnou pokračovat tím, že vynulujeme první řádek a první sloupce vyjma pozice $A_{(1,1)}$. Poté postupujeme analogicky na podmatici počínaje prvkem $a_{2,2}$. Tímto nakonec dostaneme diagonální matici D, která je Smithovým tvarem matice A.

2) Veškeré elementární operace na matici A byly prováděny přenásobením matice A zprava maticemi R_i anebo zleva maticemi L_j. Nechť počet R_i je roven n a počet L_j roven m, potom $R = R_1R_2...R_n$, $L = L_mL_{m-1}...L_1$ a $D = LAR$.

3) Abychom získali řešení soustavy $Ax = b$, spočteme $c = Lb$, tím dostaneme novou pravou stranu a vyřešíme jednoduchou diagonální soustavu $Dy = c$ a homogenní soustavu $Dt = 0$. Hledané řešení je pak $x = Ry + \langle Rt \rangle$.

6.2.2 Demonstrace algoritmu na příkladu

Nechť máme zadanou soustavu rovnic nad Z_{p^k}, $p = 3$, $k = 2$, tvaru $Ax = b$, kde

$$A = \begin{pmatrix} 2 & 7 & 8 \\ 0 & 5 & 3 \\ 3 & 2 & 0 \end{pmatrix}, b = \begin{pmatrix} 2 \\ 3 \\ 0 \end{pmatrix}.$$

1) Prvek na pozici $a_{11} = 2 = 2 \cdot 3^0$, má 3-valuaci rovnou 0, najdeme tedy multiplikativní inverz b k prvku $a_{11} = 2$, t.j., $b = 5$. První řádek vynásobíme prvkem $b = 5$ a to přenásobením matice A zleva maticí $D_1(5) = \begin{pmatrix} 5 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} := L_1$ a dostaneme matici $A_1 = \begin{pmatrix} 1 & 8 & 4 \\ 0 & 5 & 3 \\ 3 & 2 & 0 \end{pmatrix}$.

2) Nyní provedeme elementární úpravy na matici A_1, aby prvky a_{12}, a_{13}, a_{31} se rovnaly nule. Tedy přenásobíme matici A_1 zprava maticí $T_{12}(1) = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} :=$

R_1, výslednou matici pak přenásobíme zprava maticí $T_{13}(5) = \begin{pmatrix} 1 & 0 & 5 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} :=$ R_2, výslednou matici přenásobíme zleva maticí $T_{31}(6) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 6 & 0 & 1 \end{pmatrix} := L_2$ a dostaneme matici $A_2 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 5 & 3 \\ 0 & 5 & 6 \end{pmatrix}$.

3) Prvek $a_{22} = 5 = 5 \cdot 3^0$ má 3-valuaci rovnou nule, najdeme tedy multiplikativní inverz b k prvku $a_{22} = 5$, t.j, $b = 2$. Druhý řádek vynásobíme prvkem $b = 2$ a to přenásobením matice A_2 zleva maticí $D_2(2) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{pmatrix} := L_3$ a dostaneme matici $A_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 6 \\ 0 & 5 & 6 \end{pmatrix}$.

4) Vynulujeme prvky $a_{23} = 6$ a $a_{32} = 5$ a to přenásobením matice A_3 zprava maticí $T_{23}(3) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{pmatrix} := R_3$ a výslednou matici přenásobíme zleva maticí $T_{32}(4) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 4 & 1 \end{pmatrix} := L_4$ a dostaneme $A_4 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{pmatrix} := D$, která je hledaný Smithův tvar matice A.

5) Nechť tedy $R = R_1R_2R_3 = \begin{pmatrix} 1 & 1 & 8 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{pmatrix}$ a $L = L_4L_3L_2L_1 = \begin{pmatrix} 5 & 0 & 0 \\ 0 & 2 & 0 \\ 3 & 8 & 1 \end{pmatrix}$, potom $LAR = D = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 3 \end{pmatrix}$.

6) Nyní vyjádříme $c = Lb = \begin{pmatrix} 1 \\ 6 \\ 3 \end{pmatrix}$ a vyřešíme diagonální soustavu rovnic tvaru $Dy = c$, odkud získáme matici $y = \begin{pmatrix} 1 \\ 6 \\ 1 \end{pmatrix}$. Řešením homogenní soustavy $Dy = 0$ je lineární obal vektoru $\begin{pmatrix} 0 \\ 0 \\ 3 \end{pmatrix}$. Hledaným řešením je pak $x = Ry = \begin{pmatrix} 6 \\ 0 \\ 1 \end{pmatrix} + \left\langle \begin{pmatrix} 6 \\ 0 \\ 3 \end{pmatrix} \right\rangle$.

6.2.3 Implementace algoritmu v programu Mathematica

```
(*funkce Tm, Pm, Dm provadeji elementarni operace*)
Tm[matice_, index1_, index2_, strana_, radku_, sloupcu_, prvek_,
   mo_] := Module[{mat, Tmat, i, j, str, rad, sloup, b, m},
   mat = matice; i = index1; j = index2; str = strana; rad = radku;
   sloup = sloupcu; b = prvek; m = mo;
   If[str == 1, Tmat = IdentityMatrix[rad];
    Tmat[[i, j]] = (Tmat[[i, j]] + b); mat = Tmat.mat; L = Tmat.L;
    L = Mod[L, m], Tmat = IdentityMatrix[sloup];
    Tmat[[i, j]] = (Tmat[[i, j]] + b); mat = mat.Tmat; R = R.Tmat;
    R = Mod[R, m]];
   Mod[mat, m]
   ];

Pm[matice_, index1_, index2_, strana_, radku_, sloupcu_, mo_] :=
  Module[{mat, Pmat, i, j, str, rad, sloup, m},
   mat = matice; i = index1; j = index2; str = strana; rad = radku;
   sloup = sloupcu; m = mo;
   If[str == 1, Pmat = IdentityMatrix[rad];
    Pmat[[i, i]] = (Pmat[[i, i]] - 1);
    Pmat[[j, j]] = (Pmat[[j, j]] - 1);
    Pmat[[i, j]] = (Pmat[[i, j]] + 1);
    Pmat[[j, i]] = (Pmat[[j, i]] + 1); mat = Pmat.mat; L = Pmat.L;
    L = Mod[L, m], Pmat = IdentityMatrix[sloup];
    Pmat[[i, i]] = (Pmat[[i, i]] - 1);
    Pmat[[j, j]] = (Pmat[[j, j]] - 1);
    Pmat[[i, j]] = (Pmat[[i, j]] + 1);
    Pmat[[j, i]] = (Pmat[[j, i]] + 1); mat = mat.Pmat; R = R.Pmat;
    R = Mod[R, m]];
   Mod[mat, m]
   ];

Dm[matice_, index_ , strana_ , radku_, sloupcu_, prvek_, mo_] :=
  Module[{mat, Dmat, i, str, rad, sloup, b, m},
   mat = matice; i = index; str = strana; rad = radku;
   sloup = sloupcu; b = prvek; m = mo;
   If[str == 1, Dmat = IdentityMatrix[rad];
    Dmat[[i, i]] = (Dmat[[i, i]] + (b - 1)); mat = Dmat.mat;
    L = Dmat.L; L = Mod[L, m], Dmat = IdentityMatrix[sloup];
    Dmat[[i, i]] = (Dmat[[i, i]] + (b - 1)); mat = mat.Dmat;
    R = R.Dmat; R = Mod[R, m]];
   Mod[mat, m]
   ];

(*kontoluje spravny format vstupnich matic a zda se jedna \
 o prvocislo*)
Test[matice1_, matice2_, prvocislo_] := Module[{mat1, mat2, prv},
   mat1 = matice1; mat2 = matice2; prv = prvocislo;
   If[MatrixQ[A] == False || MatrixQ[B == False], e = False;
    Print["Bad matrix form"],
    If[PrimeQ[prv] == False, e = False; Print[prv "is not prime"],
     If[Length[mat1[[1]]] != Length[mat2], e = False;
      Print["Bad matrix shape"], e = True]]];
   e
   ];
```

```
(*vrati p-valuaci cisla n*)
pVal[cislo_, prvocislo_] := Module[{n, p, val, zbytek},
   n = cislo; p = prvocislo; val = 0; zbytek = 0;
   If[p > 1,
    While[zbytek == 0 && n != 0, zbytek = Mod[n, p];
     If[zbytek == 0, val++; n = (n/p)]]];
   val
   ];

(*najde ve vstupni matici  prvek s nejmensi p-valuaci, da ho na \
pozici (i,j) a elem. operacemi prevede na p^p-val*)
Nej[matice_, index1_, index2_, radku_, sloupcu_, prvocislo_, mo_] :=
  Module[{mat, i, j, rad, sloup, a, pozR, pozS, vysl, m, n, prv, val,
    moo, c, cinv},
   mat = matice; i = index1; j = index2; rad = radku; sloup = sloupcu;
    pozR = i; pozS = j; vysl = False; m = i; n = j; prv = prvocislo;
   moo = mo;
   If[mat[[i, j]] == 0,
    While[vysl == False && (m <= rad && n <= sloup),
     If[mat[[m, n]] != 0, a = pVal[mat[[m, n]], prv]; pozR = m;
      pozS = n; vysl = True]; If[n == sloup, n = j; m++, n++]],
    a = pVal[mat[[i, j]], prv]; vysl = True];
   If[vysl == True,
    For[k = i, k <= rad, k++,
     For[l = j, l <= sloup, l++,
      If[(pVal[mat[[k, l]], prv] < a) && (mat[[k, l]] != 0),
       a = pVal[mat[[k, l]], prv]; pozR = k ; pozS = l]]]];
   If[vysl == True, mat = Pm[mat, i, pozR, 1, rad, sloup, moo];
    mat = Pm[mat, j, pozS, 0, rad, sloup, moo]; mat = Mod[mat, moo];
    val = pVal[mat[[i, j]], prv]; c = (mat[[i, j]]/(prv^val));
    c = Mod[c, moo];
    If[c != 0, cinv = PowerMod[c, -1, moo];
     mat = Dm[mat, i, 1, rad, sloup, cinv, moo]], er = "err"];
   Mod[mat, moo]
   ];

(*vynuluje i-ty radek a j-ty sloupec vyjma pozice (i,j)*)
Nul[matice_, index1_, index2_, radku_, sloupcu_, mo_] :=
  Module[{mat, i, j, rad, sloup, d, m},
   mat = matice; rad = radku; sloup = sloupcu; i = index1;
   j := index2; d = 1; m = mo;
   For[k = (j + 1), k <= sloup, k++,
    If[mat[[i, k]] != 0, d = (mat[[i, k]]/mat[[i, j]]);
     mat = Tm[mat, i, k, 0, rad, sloup, (-d), m]]];
   For[l = (i + 1), l <= rad, l++,
    If[mat[[l, j]] != 0, d = (mat[[l, j]]/mat[[i, j]]);
     mat = Tm[mat, l, j, 1, rad, sloup, (-d), m]]];
   Mod[mat, m]
   ];

(*najde Smithuv tvar vstupni matice*)
SNF[matice_, radku_, sloupcu_, prvocislo_, mo_] :=
  Module[{mat, rad, sloup, i, j, vysl, prv, m},
   mat = matice; rad = radku; sloup = sloupcu; i = 1; j = 1;
   prv = prvocislo; m = mo;
   While[i <= rad && j <= sloup && er != "err",
    mat = Nej[mat, i, j, rad, sloup, prv, m];
    If[er != "err", mat = Nul[mat, i, j, rad, sloup, m]];
```

```
    i++; j++];
   Mod[mat, m]
   ];

(*resi kongruencni rovnice*)
Rovnice[prom1_, prom2_, modul_] :=
  Module[{p1, p2, mo, nalezeno, i, vysledek},
   p1 = prom1; p2 = prom2; mo = modul; nalezeno = False;
   vysledek = Reduce[p1 x == p2, {x}, Modulus -> mo];
   vysledek = ToString[vysledek];
   If[vysledek == "False", nalezeno = False,
    vysledek = StringDrop[vysledek, 5];
    vysledek = ToExpression[vysledek];
    If[Length[vysledek] > 1, vysledek = vysledek[[1]]];
    nalezeno = True];
   If[nalezeno == True, vysledek, -1]
   ];

(*resi soustavu rovnic nad Z_p^k*)
SolveZpk[matice1_, matice2_, prvocislo_, mocnina_] :=
  Module[{mat1, mat2, c, y, ynull, rad, sloup, diag, solve, j, i, prv,
     moc, m, val, hom, hodnostD, h, ch},
   prv = prvocislo; mat1 = matice1; mat2 = matice2;
   rad = Length[mat1]; sloup = Length[mat1[[1]]];
   R = IdentityMatrix[sloup]; L = IdentityMatrix[rad];
   e = Test[mat1, mat2, prv]; er = "o";
   If[e == True, moc = mocnina; m = prv^moc;
    y = Table[0, {sloup}, {1}];
    diag = SNF[mat1, rad, sloup, prv, m];
    c = L.mat2; c = Mod[c, m] ; hodnostD = MatrixRank[diag];
    For[a = hodnostD + 1, a <= rad, a++,
     If[c[[a, 1]] != 0, e = False]];
    If[e == True, h = 1; ch = 1;
     While[h <= hodnostD && e == True,
      g = Rovnice[diag[[h, ch]], c[[h, 1]], m];
      If[g == -1, e = False, y[[h, 1]] = g]; h++; ch++]];
    If[e == True, j = 1; i = 1; hom = Table[0, {0}, {0}];
     While[i <= rad && j <= sloup,
      If[diag[[i, j]] > 1, ynull = Table[0, {sloup}, {1}];
       val = pVal[diag[[i, j]], prv]; ynull[[i, 1]] = prv^(moc - val);
        ynull = Mod[R.ynull, m]; hom = Prepend[hom, ynull]];
      If[diag[[i, j]] == 0, ynull = Table[0, {sloup}, {1}];
       ynull[[i, 1]] = 1; ynull = R.ynull; hom = Prepend[hom, ynull]];
       j++; i++]], e = False];
   If[e == True, solve = R.y; solve = Mod[solve, m]; Print[solve];
    Print[hom], Print["No solution over Zp"]];
   ];
```

Ukázka programu na příkladu nad Z_{2^7}:

Nechť máme zadanou soustavu lineárních rovnic tvaru $Ax = b$, kde x je sloupcový vektor řešení dané soustavy, potom:

Vstup: $A = \{\{115, 35, 250\}, \{15, 300, 0\}, \{320, 45, 80\}\}; b = \{\{88\}, \{52\}, \{28\}\};$

Zavolání programu: SolveZpk[A, b, 2, 7]

Výstup: {{28}, {12}, {16}}

{{{0}, {0}, {64}}}

Výstupem programu je sloupcový vektor $\begin{pmatrix} x_1 = \{28\} \\ x_2 = \{12\} \\ x_3 = \{16\} \end{pmatrix}$, který odpovídá nehomogennímu řešení dané soustavy a na dalším řádku je lineární obal sloupcového vektoru $\begin{pmatrix} 0 \\ 0 \\ 64 \end{pmatrix}$, který odpovídá řešení homogenní soustavy $Ax = 0$.

6.3 $Z_{p_1^{k_1} p_2^{k_2} ... p_n^{k_n}}$

6.3.1 Popis algoritmu

Nechť máme zadanou soustavu rovnic nad Z_m, $m = p_1^{k_1} p_2^{k_2} \dots p_n^{k_n}$, tvaru $Ax = b$.

1) Dle Důsledku 1 je $Z_m \cong Z_{p_1^{k_1}} \times Z_{p_2^{k_2}} \times ... \times Z_{p_n^{k_n}}$. Užitím Věty 6 můžeme řešit zadanou soustavu zvlášť pro každý okruh $Z_{p_i^{k_i}}$, pro všechna $i = 1, ..., n$ a to algoritmem popsaným v podkapitole pro okruh Z_{p^k}.

2) Z výsledků jednotlivých okruhů pak užitím Věty 4 získáme řešení v původním okruhu Z_m.

6.3.2 Demonstrace algoritmu na příkladu

Nechť máme zadanou soustavu rovnic nad Z_m, $m = 12 = 2^2 * 3$, tvaru $Ax = b$, kde

$$A = \begin{pmatrix} 2 & 4 & 6 \\ 3 & 9 & 7 \end{pmatrix}, b = \begin{pmatrix} 2 \\ 0 \end{pmatrix}.$$

Soustavu tedy nejprve vyřešíme nad okruhem Z_{2^2}, poté nad okruhem Z_3 a z výsledků použitím Věty 4 získáme řešení v okruhu Z_{12}. K získání výsledků v okruzích Z_{2^2} a Z_3 použijeme již popsaný algoritmus pro Z_{p^k}.

1) V Z_{2^2} získáme řešení $\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + \left\langle \begin{pmatrix} 2 \\ 2 \\ 0 \end{pmatrix} \right\rangle$.

2) V Z_3 získáme řešení $\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$.

3) Využijeme Větu 4 a vyřešíme následné soustavy kongruencí:

$x_1 \equiv 1 \ mod\ 4$	$x_2 \equiv 1 \ mod\ 4$	$x_3 \equiv 0 \ mod\ 4$
$x_1 \equiv 1 \ mod\ 3$	$x_2 \equiv 0 \ mod\ 3$	$x_3 \equiv 0 \ mod\ 3$

Tedy $x_1 \equiv 1 \ mod\ 12$, $x_2 \equiv 9 \ mod\ 12$, $x_3 \equiv 0 \ mod\ 12$. Řešení pro homogenní soustavu se v tomto případě zachová, takže dostáváme celkové řešení pro Z_{12}

$$\begin{pmatrix} 1 \\ 9 \\ 0 \end{pmatrix} + \left\langle \begin{pmatrix} 2 \\ 2 \\ 0 \end{pmatrix} \right\rangle.$$

6.3.3 Implementace algoritmu v programu Mathematica

```
(*funkce Tm, Pm, Dm provadeji elementarni operace*)
Tm[matice_, index1_, index2_, strana_, radku_, sloupcu_, prvek_,
   mo_] := Module[{mat, Tmat, i, j, str, rad, sloup, b, m},
   mat = matice; i = index1; j = index2; str = strana; rad = radku;
   sloup = sloupcu; b = prvek; m = mo;
   If[str == 1, Tmat = IdentityMatrix[rad];
```

```
    Tmat[[i, j]] = (Tmat[[i, j]] + b); mat = Tmat.mat;
    L = Mod [Tmat.L, m], Tmat = IdentityMatrix[sloup];
    Tmat[[i, j]] = (Tmat[[i, j]] + b); mat = mat.Tmat;
    R = Mod[R.Tmat, m]];
   Mod[mat, m]
   ];

Pm[matice_, index1_, index2_, strana_, radku_, sloupcu_, mo_] :=
  Module[{mat, Pmat, i, j, str, rad, sloup, m},
   mat = matice; i = index1; j = index2; str = strana;
   rad = radku; sloup = sloupcu; m = mo;
   If[str == 1, Pmat = IdentityMatrix[rad];
    Pmat[[i, i]] = (Pmat[[i, i]] - 1);
    Pmat[[j, j]] = (Pmat[[j, j]] - 1);
    Pmat[[i, j]] = (Pmat[[i, j]] + 1);
    Pmat[[j, i]] = (Pmat[[j, i]] + 1); mat = Pmat.mat;
    L = Mod[Pmat.L, m], Pmat = IdentityMatrix[sloup];
    Pmat[[i, i]] = (Pmat[[i, i]] - 1);
    Pmat[[j, j]] = (Pmat[[j, j]] - 1);
    Pmat[[i, j]] = (Pmat[[i, j]] + 1);
    Pmat[[j, i]] = (Pmat[[j, i]] + 1); mat = mat.Pmat;
    R = Mod[R.Pmat, m]];
   Mod[mat, m]
   ];

Dm[matice_, index_ , strana_ , radku_, sloupcu_, prvek_, mo_] :=
  Module[{mat, Dmat, i, str, rad, sloup, b, m},
   mat = matice; i = index; str = strana; rad = radku;
   sloup = sloupcu; b = prvek; m = mo;
   If[str == 1, Dmat = IdentityMatrix[rad];
    Dmat[[i, i]] = (Dmat[[i, i]] + (b - 1)); mat = Dmat.mat;
    L = Mod[Dmat.L, m], Dmat = IdentityMatrix[sloup];
    Dmat[[i, i]] = (Dmat[[i, i]] + (b - 1)); mat = mat.Dmat;
    R = Mod[R.Dmat, m]];
   Mod[mat, m]
   ];

(*kontoluje spravny format vstupnich matic a zda se jedna \
 o prvocislo*)
Test[matice1_, matice2_] := Module[{mat1, mat2},
   mat1 = matice1; mat2 = matice2;
   If[MatrixQ[A] == False || MatrixQ[B == False], e = False;
    Print["Bad matrix form"],
    If[Length[mat1] != Length[mat2], e = False;
     Print["Bad matrix shape"], e = True]];
   e
   ];

(*vrati p-valuaci cisla n*)
pVal[cislo_, prvocislo_] := Module[{n, p, val, zbytek},
   n = cislo; p = prvocislo; val = 0; zbytek = 0;
   If[p > 1,
    While[zbytek == 0 && n != 0, zbytek = Mod[n, p];
     If[zbytek == 0, val++; n = (n/p)]]];
   val
   ];
```

```
(*najde ve vstupni matici  prvek s nejmensi p-valuaci, da ho na \
pozici (i,j) a elem. operacemi prevede na p^p-val*)
Nej[matice_, index1_, index2_, radku_, sloupcu_, prvocislo_, mo_] :=
  Module[{mat, i, j, rad, sloup, a, pozR, pozS, vysl, m, n, prv, val,
    moo, c, cinv},
   mat = matice; i = index1; j = index2; rad = radku; sloup = sloupcu;
    pozR = i; pozS = j; vysl = False; m = i; n = j; prv = prvocislo;
   moo = mo;
   If[mat[[i, j]] == 0,
    While[vysl == False && (m <= rad && n <= sloup),
     If[mat[[m, n]] != 0, a = pVal[mat[[m, n]], prv]; pozR = m;
      pozS = n; vysl = True]; If[n == sloup, n = j; m++, n++]],
    a = pVal[mat[[i, j]], prv]; vysl = True];
   If[vysl == True,
    For[k = i, k <= rad, k++,
     For[l = j, l <= sloup, l++,
      If[(pVal[mat[[k, l]], prv] < a) && (mat[[k, l]] != 0),
       a = pVal[mat[[k, l]], prv]; pozR = k ; pozS = l]]]];
   If[vysl == True, mat = Pm[mat, i, pozR, 1, rad, sloup, moo];
    mat = Pm[mat, j, pozS, 0, rad, sloup, moo]; mat = Mod[mat, moo];
    val = pVal[mat[[i, j]], prv]; c = (mat[[i, j]]/(prv^val));
    c = Mod[c, moo];
    If[c != 0, cinv = PowerMod[c, -1, moo];
     mat = Dm[mat, i, 1, rad, sloup, cinv, moo]], er = "err"];
   Mod[mat, moo]
   ];

(*vynuluje i-ty radek a j-ty sloupec vyjma pozice (i,j)*)
Nul[matice_, index1_, index2_, radku_, sloupcu_, mo_] :=
  Module[{mat, i, j, rad, sloup, d, m},
   mat = matice; rad = radku; sloup = sloupcu; i = index1;
   j := index2; d = 1; m = mo;
   For[k = (j + 1), k <= sloup, k++,
    If[mat[[i, k]] != 0, d = (mat[[i, k]]/mat[[i, j]]);
     mat = Tm[mat, i, k, 0, rad, sloup, (-d), m]]];
   For[l = (i + 1), l <= rad, l++,
    If[mat[[l, j]] != 0, d = (mat[[l, j]]/mat[[i, j]]);
     mat = Tm[mat, l, j, 1, rad, sloup, (-d), m]]];
   Mod[mat, m]
   ];

(*najde Smithuv tvar vstupni matice*)
SNF[matice_, radku_, sloupcu_, prvocislo_, mo_] :=
  Module[{mat, rad, sloup, i, j, vysl, prv, m},
   mat = matice; rad = radku; sloup = sloupcu; i = 1; j = 1;
   prv = prvocislo; m = mo;
   While[i <= rad && j <= sloup && er != "err",
    mat = Nej[mat, i, j, rad, sloup, prv, m];
    mat = Nul[mat, i, j, rad, sloup, m];
    i++; j++];
   Mod[mat, m]
   ];

(*resi kongruencni rovnice*)
Rovnice[prom1_, prom2_, modul_] :=
  Module[{p1, p2, mo, nalezeno, i, vysledek},
   p1 = prom1; p2 = prom2; mo = modul; nalezeno = False;
   vysledek = Reduce[p1 x == p2, {x}, Modulus -> mo];
```

```
    vysledek = ToString[vysledek];
    If[vysledek == "False", nalezeno = False,
     vysledek = StringDrop[vysledek, 5];
     vysledek = ToExpression[vysledek];
     If[Length[vysledek] > 1, vysledek = vysledek[[1]]];
     nalezeno = True];
    If[nalezeno == True, vysledek, -1]
    ];

(*resi soustavu rovnic nad Z_p^k*)
SolveZpk[matice1_, matice2_, prvocislo_, mocnina_] :=
  Module[{mat1, mat2, c, y, ynull, rad, sloup, diag, solve, j, i, prv,
    moc, m, val, hom, hodnostD, h, ch},
   prv = prvocislo; er = "o"; mat1 = matice1; mat2 = matice2;
   rad = Length[mat1]; sloup = Length[mat1[[1]]]; moc = mocnina;
   m = prv^moc; y = Table[0, {sloup}, {1}];
   diag = SNF[mat1, rad, sloup, prv, m];
   c = L.mat2; c = Mod[c, m] ; hodnostD = MatrixRank[diag];
   For[a = hodnostD + 1, a <= rad, a++, If[c[[a, 1]] != 0, e = False]];
   If[e == True, h = 1; ch = 1;
    While[h <= hodnostD && e == True,
     g = Rovnice[diag[[h, ch]], c[[h, 1]], m];
     If[g == -1, e = False, y[[h, 1]] = g]; h++; ch++]];
   If[e == True, solve = R.y; solve = Mod[solve, m]];
   If[e == True, j = 1; i = 1; hom = Table[0, {0}, {0}];
    While[i <= rad && j <= sloup,
     If[diag[[i, j]] > 1, ynull = Table[0, {sloup}, {1}];
      val = pVal[diag[[i, j]], prv]; ynull[[i, 1]] = prv^(moc - val);
      ynull = Mod[R.ynull, m]; hom = Prepend[hom, ynull]];
     If[diag[[i, j]] == 0, ynull = Table[0, {sloup}, {1}];
      ynull[[i, 1]] = 1; ynull = R.ynull; hom = Prepend[hom, ynull]];
     j++; i++]];
   If[e == True, {solve, hom}, {}]
   ];

(*prevadi jednotliva reseni v Z_p^k do reseni v Z_m*)
Lagrange[moduly_, zbytky_, zaklad_] :=
  Module[{modul, m, u, n, r, delka, vysledek},
   m = zaklad; modul = moduly; u = zbytky; delka = Length[modul];
   vysledek = 0;
   n = Table[1, {delka}]; r = Table[1, {delka}];
   For[i = 1, i <= delka, i++, n[[i]] = (m/modul[[i]]);
    r[[i]] = PowerMod[n[[i]], -1, modul[[i]]]];
   For[i = 1, i <= delka, i++,
    vysledek = (vysledek + r[[i]]*n[[i]]*u[[i]])];
   Mod[vysledek, m]
   ];

(*resi soustavu rovnic nad Z_m*)
SolveZm[matice1_, matice2_, zaklad_] :=
  Module[{mat1, mat2, rad, sloup, zak, rozklad, delka, seznam1,
    seznam2, vypocet, i, poz1, poz2, okruh, zbytk, x, hom, Hom,
    DelkaHom, DelkaSloup, zbytkHom, LZ, q},
   mat1 = matice1; mat2 = matice2; zak = zaklad; rad = Length[mat1];
   sloup = Length[mat1[[1]]]; R = IdentityMatrix[sloup];
   L = IdentityMatrix[rad]; e = Test[mat1, mat2];
   rozklad = FactorInteger[zaklad]; delka = Length[rozklad];
   seznam1 = Table[0, {0}, {0}]; seznam2 = Table[0, {0}, {0}];
```

```
okruh = Table[1, {delka}];
x = Table[0, {sloup}, {1}]; zbytk = Table[1, {delka}];
zbytkHom = Table[0, {delka}]; DelkaHom = 0;
Hom = Table[0, {0}, {0}];
i = 1;
While[i <= delka && e == True, R = IdentityMatrix[sloup];
 L = IdentityMatrix[rad]; poz1 = rozklad[[i, 1]];
 poz2 = rozklad[[i, 2]];
 vypocet = SolveZpk[mat1, mat2, poz1, poz2];
 If[vypocet != {}, seznam1 = Append[seznam1, vypocet[[1]]];
  If[vypocet[[2]] == {},
   seznam2 = Append[seznam2, {Table[0, {sloup}, {1}]}],
   seznam2 = Append[seznam2, vypocet[[2]]]];
  If[Length[vypocet[[2]]] > DelkaHom,
   DelkaHom = Length[vypocet[[2]]], e = False]; i++];
If[e == True,
 For[j = 1, j <= delka, j++,
  okruh[[j]] = rozklad[[j, 1]]^rozklad[[j, 2]]];
 For[k = 1, k <= sloup, k++,
  For[l = 1, l <= delka, l++, zbytk[[l]] = seznam1[[l, k]]];
  x[[k, 1]] = Lagrange[okruh, zbytk, zak]];
 DelkaSloup = 0;
 For[p = 1, p <= delka, p++, DelkaSloup = Length[seznam2[[p]]];
  If[DelkaSloup < DelkaHom,
   seznam2[[p]] = Append[seznam2[[p]], Table[0, {sloup}, {1}]]]];
 LZ = 1; q = 1;
 While[LZ <= sloup && q <= DelkaHom, hom = Table[0, {sloup}, {1}];
  For[r = 1, r <= sloup, r++,
   For[s = 1, s <= delka, s++, zbytkHom[[s]] = seznam2[[s, q, r]]];
    hom[[r, 1]] = Lagrange[okruh, zbytkHom, zak]];
  Hom = Append[Hom, hom]; LZ++; q++]; Print[x]; Print[Hom],
 Print["No solution over Zm"]];
];
```

Návrh na funkci *Lagrange* převzat z [9, podkapitola 2.7.1]

Ukázka programu na příkladu nad $Z_{1210104}$:

Nechť máme zadanou soustavu lineárních rovnic tvaru $Ax = b$, kde x je sloupcový vektor řešení dané soustavy, potom:

Vstup: $A = \{\{2, 4, 6\}, \{3, 9, 7\}\}; b = \{\{24\}, \{58\}\};$

Zavolání programu: SolveZm[A, b,1210104]

Výstup: {{1120464}, {851562}, {672280}}
{{{605052}, {605052}, {0}}}

Výstupem programu je sloupcový vektor $\begin{pmatrix} x_1 = \{1120464\} \\ x_2 = \{851562\} \\ x_3 = \{672280\} \end{pmatrix}$, který odpovídá nehomogennímu řešení dané soustavy a na dalším řádku je lineární obal sloup-

cového vektoru $\begin{pmatrix} 605052 \\ 605052 \\ 0 \end{pmatrix}$, který odpovídá řešení homogenní soustavy $Ax = 0$.

Závěr

V práci byl popsán způsob, jakým se dají řešit soustavy lineárních rovnic nad okruhy hlavních ideálů. Konkrétně byl představen algoritmus pro řešení soustav lineárních rovnic nad okruhy $Z, Q[x]$ a Z_m. U každého z těchto okruhů byl algoritmus slovně popsán, následovala ukázka algoritmu na příkladu a jeho možná implementace v programu Mathematica.

Jiné metody, které je možné využít pro získání Smithova normálního tvaru pro zadanou matici, lze pro okruh celých čísel nalézt v práci Havas a Majewski, 1997 [4], pro polynomiální matice v práci Storjohann a Labahn, 1994 [10] a pro Z_m v práci Hartung, 2010 [3].

Seznam použité literatury

[1] ANDERSON, Frank W. *Rings and Categories of Modules*. 2rd edition, 1992. New York: Springer-Verlag Inc. 376 p. ISBN 0-387-97845-3.

[2] DUMMIT, David S., FOOTE, Richard M. *Abstract Algebra*. 3rd edition, 2004. USA: Malloy Inc. 932 p. ISBN 0-471-43334-9.

[3] HARTUNG, René. *Solving linear equations over finitely generated abelian groups*. Cornell University Library [online]. July 2010. [cit. 2012-07-20]. Dostupné z <http://arxiv.org/abs/1007.2477v1>.

[4] HAVAS, George, MAJEWSKI, Bohdan S. *Integer Matrix Diagonalization*. University of Queensland [online]. 1997. [cit. 2012-07-29]. Dostupné z <http://itee.uq.edu.au/ havas/1997hm.pdf>.

[5] JACOBSON, Nathan. *Basic Algebra I*. 1st edition, 1974. San Francisco: W. H. Freeman and Company. 472 p. ISBN 0716704536.

[6] MACLANE, S., BIRKHOFF, G. *Algebra*. 2. vyd. Bratislava: Alfa, 1974. 664 s. ISBN 63-551-74.

[7] SHASTRI, Parvati. *Lectures on Modules over Principal Ideal Domains*. University of Mumbai [online]. [cit. 2012-07-20]. Dostupné z <http://www.math.iitb.ac.in/atm/final1.pdf>.

[8] STANOVSKÝ, David. *Základy algebry*. 1. vyd. Praha: Matfyzpress, 2010. 153 s. ISBN 978-80-7378-105-7.

[9] STANOVSKÝ, David, BARTO, Libor. *Počítačová algebra*. 1. vyd. Praha: Matfyzpress, 2011. 180 s. ISBN 978-80-7378-167-5.

[10] STORJOHANN, Arne, LABAHN, George. *A Fast Las Vegas Algorithm for Computing the Smith Normal Form of a Polynomial Matrix*. University of Waterloo [online]. November 1994. [cit. 2012-07-29]. Dostupné z <http://www.cs.uwaterloo.ca/research/tr/1994/43/CS-94-43.pdf>.