Fakulta informatiky Masarykovy Univerzity
v Brně

Diplomová práce

Systémy počítačové algebry a algebra

Algoritmy na výpočet největšího společného dělitele

Brno, duben 1997

Michal Bulant

Vedoucí DP: Doc. RNDr. Jiří Hřebíček, CSc.

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a použil pouze uvedené literatury.

V Brně 12. 4. 1997 Michal Bulant

Věnováno Janě a Alence

Obsah

Kapitola 1

Úvod

Systémy počítačové algebry se dostávají zejména v poslední době do centra pozornosti matematiků a odborníků v oborech svázaných s matematikou. Je to dáno zejména tím, že narozdíl od systémů numerických, mohou tyto systémy operovat nad symbolickými matematickými objekty a provádět tyto operace nad rozsáhlými strukturami bez nutnosti zaokrouhlování či omezení velikosti. Systémy disponují rozsáhlou škálou symbolických schopností — od manipulací s polynomy, mocninnými řadami a racionálními funkcemi přes symbolické derivování a integrování až po řešení polynomiálních, rekurentních či diferenciálních rovnic. Přitom jejich výhodou oproti postupu „tužka a papír“ je zejména rychlost, paměťová kapacita a rovněž velké množství matematické inteligence, kterou tyto systémy obsahují ve formě zabudovaných procedur.

V této práci se po stručné zmínce o historii systémů počítačové algebry ve 2. kapitole, kde se zaměříme na v současnosti zřejmě nejrozšířenější systém **Maple** a na velmi mocný systém (zatím ne příliš rozšířený) **Axiom**, budeme věnovat algoritmům na výpočet největšího společného dělitele. Je to z toho důvodu, že tyto algoritmy jsou klíčové pro efektivitu těchto systémů a protože s růstem nároků na velikost objektů, se kterými jsou nuceni matematikové počítat, roste požadavek na kvalitu používaných algoritmů. Algoritmus na výpočet největšího společného dělitele je pak klíčovým zejména při operacích s polynomy a racionálními funkcemi — kvůli paměťovým nárokům je totiž nutno racionální funkce vnitřně reprezentovat jako podíl polynomů ve zkrácené formě, což vyžaduje výpočet největšího společného dělitele polynomů v podstatě při každém sčítání racionálních funkcí. Algoritmus na výpočet největšího společného dělitele musí být samozřejmostí rovněž při práci ve složitějších algebraických strukturách, jako např. při konečných rozšířeních těles, kdy je výpočet největšího společného dělitele nutný k provádění základních algebraických operací.

Ve 3. kapitole pak vybudujeme nutnou teorii pro to, abychom byli schopni prezentovat tyto algoritmy a zdůvodnit jejich správnost a efektivitu. Definujeme přitom i základní pojmy, aby byl čtenář schopen číst tuto práci bez další literatury, příliš se však nezabýváme dalšími matematickými souvislostmi, které příliš nesouvisí s posláním tohoto textu. Na druhé straně uvedeme množství odkazů na další literaturu, která se zabývá hlubší teorií okolo prezentovaných pojmů. V závěru kapitoly, kdy se specializujeme na výpočet největšího společného dělitele (GCD), uvedeme Euklidův algoritmus, který je historicky nejstarším algoritmem na výpočet GCD, a ukážeme, proč není výhodný pro polynomy s racionálními (nebo celočíselnými) koeficienty. Protože jsou právě tyto typy polynomů klíčové pro systémy počítačové algebry, vyvstala nutnost algoritmů efektivnějších.

V následující kapitole pak prezentujeme několik typů algoritmů, které jsou v současnosti používány na výpočet GCD. Při zápisu algoritmů budeme přitom používat neformálnější způsob zápisu, používaný např. D. E. Knuthem v sérii *The Art of Computer Programming*. Tvar těchto algoritmů, který je pohodlnější pro zápis v některém vyšším programovacím jazyce, je pak pro vybrané algoritmy uveden v závěrečné, 5. kapitole.

V 5. kapitole rovněž uvedené algoritmy porovnáme z hlediska efektivity a ukážeme, jak je výpočet největšího společného dělitele polynomů implementován v systémech Maple a Axiom.

V této diplomové práci vycházíme především z monografie [3], kterou doplňují přístupy z [2] a [7]. Budování teorie je založeno na přístupu z [3], který však upravíme pro naše účely a kvůli lepší srozumitelnosti textu rovněž používáme rozdílných formulací. Rovněž některá tvrzení a jejich důkazy jsou přepracovány (ať už kvůli větší názornosti nebo kvůli nepřesnostem v původním textu).

Kapitola 2

Historie systémů počítačové algebry

2.1 Počátky

Poté, co se objevily první programovací jazyky, určené převážně pro numerické výpočty (FORTRAN okolo roku 1958 a ALGOL v roce 1960), se zejména pod vlivem LISPu objevily první programy pro symbolickou manipulaci matematických výrazů. V roce 1961 to byl program SAINT (Symbolic Automatic INTegration), napsaný v LISPu na M.I.T. Tento program byl založen na množství heuristik pro neurčitou integraci a úspěšný byl podobně jako dobrý student matematické analýzy.

Jedním z prvních systémů pro symbolické výpočty byl FORMAC, vyvinutý v IBM. Jednalo se o preprocesor FORTRANu (a později PL/I), který byl určen pro manipulaci elementárních funkcí (především polynomů a racionálních funkcí). Dalšími systémy, vzniklými v první polovině šedesátých let, byly PM, jazyk pro manipulaci s polynomy, a MATHLAB, založený na LISPu, který byl prvním interaktivním systémem, sloužícím jako symbolický kalkulátor.

Druhá polovina šedesátých let se nesla především ve znamení zdokonalování systémů dřívějších. Na M.I.T. vznikl program SIN (Symbolic INTegrator), který používal narozdíl od dřívějšího SAINTu algoritmický přístup a byl rovněž mnohem efektivnější. V roce 1968 byl na Stanfordské univerzitě vyvinut REDUCE, interaktivní systém, založený na LISPu, určený pro fyzikální výpočty. Jednou z hlavních předností systému REDUCE měla být jeho přenositelnost mezi různými platformami. Nejambicióznějšími systémy konce šedesátých let pak byly SCRATCHPAD, vyvíjený v IBM Research a obsahující jako knihovny mnohé z předcházejících systémů, a MACSYMA, vyvíjená na M.I.T., která kromě obvyklých algebraických manipulací obsahovala prostředky pro výpočet limit, symbolické integrování a řešení rovnic.

V sedmdesátých letech, v důsledku většího zájmu potenciálních uživatelů, došlo k cílenějšímu vývoji algoritmů pro systémy počítačové algebry. Zejména M.I.T. vyvinul při zdokonalování algoritmů pro svůj program MACSYMA mnoho nových postupů, které jsou používány dodnes (např. modulární algoritmy). MACSYMA se tak stala v té době nejsilnějším prostředkem pro počítačovou matematiku. Prvním systémem, který byl schopen pracovat na počítačích IBM PC, byl muMATH (rovněž také μ-MATH) Hawaiské univerzity. Rozšířil tak možnost používání matematických programů pro výzkum a výuku. Kromě těchto systémů vzniklo několik programů, specializovaných na určitou oblast matematiky nebo fyziky (např. SHEEP, TRIGMAN nebo SCHOONSCHIP). V SSSR rovněž vznikl hardware pro symbolické výpočty nazvaný ANALITIK.

2.2 Moderní systémy

Kvůli vysoké náročnosti systémů počítačové algebry na hardware zůstávalo jejich používání (s výjimkou programu muMATH) omezeno na uživatele mající přístup k výkonným počítačům. Teprve v osmdesátých letech s nástupem pracovních stanic založených na mikroprocesorech bylo umožněno jejich větší rozšíření. Množství používaných operačních systémů spolu s omezeným výkonem těchto stanic si vynutily přechod k implementačním jazykům nižší úrovně, přesto však dostatečně přenositelným. Tyto požadavky splňoval především jazyk C. Toto období je charakteristické vznikem komerčních systémů.

V roce 1981 vznikl na University of Waterloo systém MAPLE, který byl navržen tak, aby splňoval kritérium široké použitelnosti i na nepříliš silných strojích, ale aby byl schopný náročných výpočtů, požadovaných vědeckými pracovníky. Toto umožňovala modulární struktura MAPLu, totiž jádro napsané v jazyku C spolu s rozsáhlou knihovnou matematických funkcí napsaných v jazyce MAPLu vyšší úrovně.

Dalším programem napsaným v jazyku C byl SMP Stephena Wolframa. Byl vysoce přenositelný, používající programovací interface založený na splňování pravidel (tzv. *pattern matching*). Z něj potom S. Wolfram se svými spolupracovníky vycházel při tvorbě systému MATHEMATICA (objevil se v roce 1988). Ve své době byl výjimečný především svými grafickými schopnostmi (2D a 3D grafika včetně animací) spolu s grafickým uživatelským prostředím.

S poněkud jinými ambicemi byl vytvořen program DERIVE, následník muMATH pro osobní počítače. Přes nepříliš bohaté možnosti symbolických výpočtů dosáhl poměrně širokého rozšíření svou orientací na jednoduché výpočty, které umožnily osobní počítače společně s příjemným uživatelským prostředím a 3D grafickými schopnostmi.

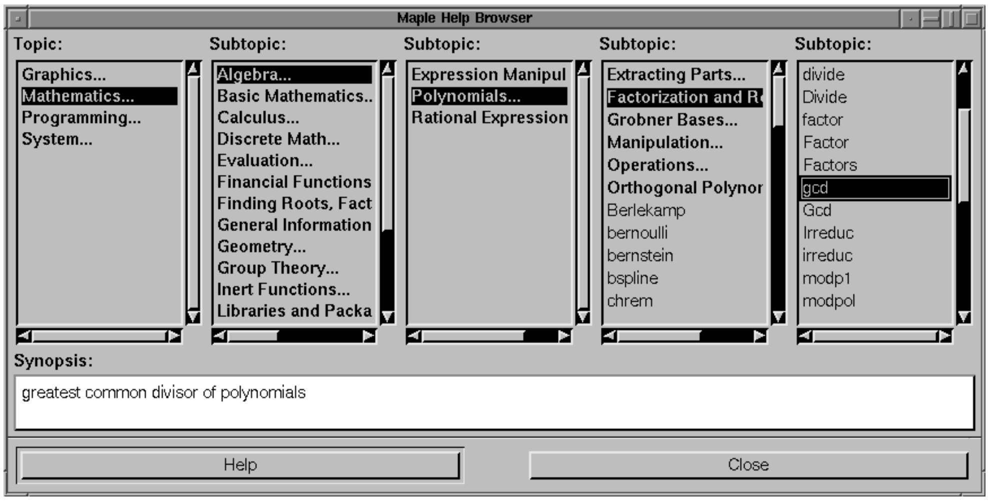
Kromě vývoje nových systému pokračovaly rovněž práce na některých starších programech. Z programu SCRATCHPAD například vycházeli tvůrci systému AXIOM (jde o část systému SCRATCHPAD II, která je vyvíjena na komerční bázi skupinou NAG[1]). AXIOM (podrobněji viz [6]) je jediným silně typovaným systémem, který umožňuje vyvíjet společné algoritmy pro různé algebraické struktury, zatímco v ostatních běžných systémech je nutné algoritmy psát pro konkrétní obor (např. zatímco v MAPLu je nutné při psaní algoritmu pro největšího společného dělitele dvou polynomů brát ohled na to, z kterého okruhu koeficienty uvažujeme, v AXIOMu je důležité pouze to, jestli příslušný okruh koeficientů je s jednoznačným rozkladem, euklidovský okruh, či těleso). Díky tomuto je AXIOM velmi mocným systémem, což však s sebou přináší, zejména v porovnání se specializovanými systémy psanými v jazyku C, mnohem větší náročnost na hardware. Rovněž nepříliš vysoká podpora různých platforem přispěla k dosud nevelkému rozšíření AXIOMu v obecném použití. Na druhou stranu, tempo růstu výkonu osobních počítačů spolu s existencí verzí AXIOMu pro nejrozšířenější operační systémy jako MS Windows a UNIX může napomoci většímu rozšíření tohoto mezi matematiky ceněného systému.

Kromě systémů obecně použitelných vzniká rovněž mnoho programů specializovaných na různá konkrétní odvětví. Díky specializaci jsou tak tyto programy zpravidla efektivnější v použití pro danou oblast. Často jsou také dostupné nekomerčními cestami. Mezi takové systémy (se specializací na algebru) patří např. PARI a KANT určené pro teorii čísel, CAYLEY (mocný komerční software, v nové verzi by měl být šířen pod jménem MAGMA) a GAP v teorii grup, LiE pro výpočty v Lieových algebrách a MAUCALAY pro komutativní algebru a algebraickou geometrii.

2.3 Maple a Axiom podrobněji

V této části se o něco podrobněji zmíníme o dvou systémech počítačové algebry, které jsou přístupné na Masarykově Universitě. Jsou jimi Maple V R4 (případně pro některé operační systémy R3) a Axiom verze 2.0. Maple je systém, který byl od počátku vytvářen se zřetelem na efektivitu a na použití v rozličných výpočetních podmínkách, proto je v současnosti pravděpodobně nejúspěšnějším komerčním systémem počítačové algebry. Naproti

[1]The Numerical Algorithms Group Ltd., také např. knihovna pro numerické výpočty

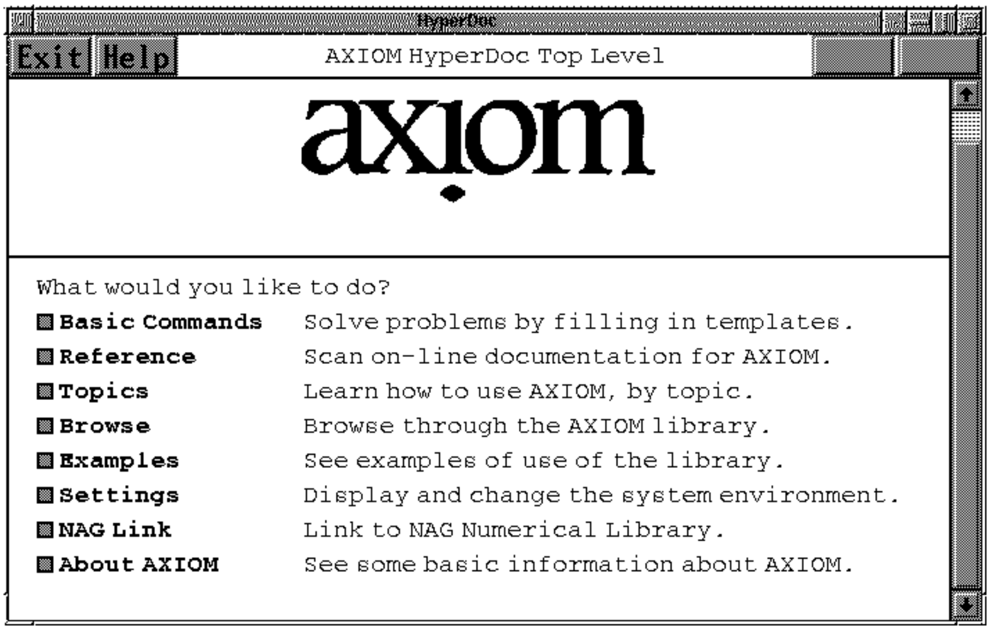


Obrázek 2.1: Systém nápovědy v Maplu, tzv. Help Browser

tomu Axiom, jak jsme uvedli v předchozí části, vychází z velmi obecně koncipovaného systému SCRATCHPAD, který byl koncipován s důrazem na co nejširší možné algebraické použití. Je proto poměrně náročný na výkon hardwaru, což ovlivňuje jeho dosavadní menší rozšířenost.

Oba systémy disponují především vlastnostmi, které jsou dnes již samozřejmostí pro všechny systémy tohoto druhu. Zvládají výpočty s libovolně velkými čísly[2], přesnou racionální aritmetiku, výpočty v pohyblivé řádové čárce s libovolnou přesností a zejména manipulace s výrazy, obsahujícími neznámé. Poslední vlastnost pak umožňuje provádět výpočty v nejrůznějších matematických strukturách, jako jsou konečná tělesa nebo algebraická číselná tělesa (jejich příkladem je těleso komplexních čísel). Systémy mají v sobě zabudováno velké množství běžných matematických funkcí a konstant. Jejich nejdůležitější předností je pak schopnost manipulovat neurčitými výrazy jako jsou polynomy, mocninné řady, racionální funkce a provádět expanzi nebo redukci výrazů pomocí dané množiny pravidel (např. všechny možné goniometrické identity). Dokáží rovněž řešit mnohé problémy z matematické analýzy jako jsou např. výpočty limit, derivací funkcí, výpočet určitého či neurčitého integrálu a řešení diferenciálních rovnic. Schopnosti těchto systémů jsou pak vhodně doplněny důležitými numerickými algoritmy, jako jsou (přibližný) výpočet kořene polynomu nebo numerická integrace. Důležitou vlast-

[2] To platí samozřejmě pouze v ideálním případě, velikost největšího možného čísla je omezena jeho vnitřní reprezentací, např. v Maplu je počet cifer největšího možného celého čísla roven $2^{19} - 9$.



Obrázek 2.2: Titulní strana hypertextové dokumentace Axiomu

ností těchto systémů je pak jejich rozšiřitelnost. Ta je dána existencí mocného programovacího jazyka, který umožňuje doplnit potřebné algoritmy.

Maple je v současnosti výhodný zejména pro své výukové a prezentační možnosti. Je schopen ve formě grafických zápisníků uchovávat informace o uskutečněných výpočtech a tyto pak kdykoli zopakovat. Na druhé straně, zřejmě na úkor velké obecnosti, však mnohdy postrádá kvalitní algoritmy pro některé operace. Použití na hlubší algebraické manipulace nad obecnými matematickými strukturami pak příliš neumožňuje (to ale také nebylo cílem vývojářů Maplu). Naproti tomu obsahuje mnohé knihovny a algoritmy, které umožňují jeho využití v oborech s matematikou spřízněných, zejména v přírodních vědách. Disponuje rovněž kvalitními možnostmi 2D a 3D grafiky včetně animací a propracovaným systémem nápovědy (viz Obrázek 2.1). Podrobnější úvodní informace je možno nalézt ve výborné učebnici [4].

Axiom je systém, který je přínosem zejména pro matematiky, kteří se zabývají teorií, neboť jeho strukturovanost a rozšiřitelnost umožňuje zkoumání mnohých oblastí matematiky, které zůstávají nad možnostmi jiných systémů. Díky tomu, že je vybaven kompilátorem a není tedy odkázán na pouhou interpretaci kódu jako většina ostatních systémů, a díky schopnosti volání funkcí implementovaných v jazyku Fortran nebo C[3] je pak zejména

[3]Axiom umožňuje například volání procedur z knihovny numerických funkcí od NAG.

u rozsáhlých výpočtů efektivnější než jiné systémy. Jeho výhoda pro matematika a programátora spočívá ve vysoké strukturovanosti matematických objektů a datových typů, kde se uplatňují mnohé principy objektového programování. Jak jsme již naznačili v části o historii, je jazyk Axiomu silně typovaný, při běžném použití Axiomu jako interaktivního kalkulátoru se však typy jednotlivých proměnných budují dynamicky podle kontextu.

Rovněž Axiom je výborně dokumentován, v systému je integrován hypertextový prohlížeč dokumentace HyperDoc, který obsahuje kromě běžných referenčních odkazů také interaktivní učebnici Axiomu a definice veškerých struktur a operací vytvořených v programovacím jazyce Axiomu (a to i těch, které jsou vytvořeny uživatelem při běhu systému).

Kapitola 3

Některé základní definice a poznatky z algebry

Jak jsme uvedli v úvodu, je výpočet největšího společného dělitele různých objektů (a zejména pak polynomů) jedním z klíčových algoritmů, potřebných pro efektivní fungování systému počítačové algebry. V této kapitole uvedeme některé důležité pojmy a poznatky z algebry, které budeme později potřebovat pro formulaci a důkazy uváděných algoritmů.

3.1 Okruhy a tělesa

Definice 3.1. *Grupou* $(G, \circ)$ [1] nazveme neprázdnou množinu G uzavřenou vzhledem k binární operaci $\circ$, takovou, že $\circ$ je asociativní operací, v G existuje jediný *neutrální* prvek e a pro každý prvek $a \in G$ existuje prvek $a^{-1} \in G$ tak, že $a \circ a^{-1} = a^{-1} \circ a = e$.

Grupu G nazveme *abelovskou* (nebo *komutativní*), je-li navíc operace $\circ$ komutativní vzhledem ke grupě G.

Okruhem $(R, +, \times)$ nazveme neprázdnou množinu R uzavřenou na dvě binární operace $+$ a $\times$, která je komutativní grupou vzhledem k operaci $+$, operace $\times$ je vzhledem k prvkům R asociativní a v R existuje vzhledem k operaci $\times$ neutrální prvek[2] (v tomto kontextu je často nazývám *jednotkový*), splňující navíc levý a pravý distributivní zákon.

[1] Běžně budeme místo dvojice $(G, \circ)$ psát pouze G, nemůže-li tak dojít k nedorozumění. Podobně v případě okruhů budeme místo $(R, +, \times)$ psát pouze symbol pro nosiče okruhu R.

[2] V obvyklých případech budeme neutrální prvek vzhledem k operaci $+$ značit 0 a neutrální prvek vzhledem k operaci $\times$ budeme značit 1.

Komutativním okruhem nazveme okruh, vzhledem ke kterému je i operace $\times$ komutativní.

Jsou-li $a, b \in R$, pak říkáme, že prvek b *dělí* prvek a, jestliže existuje $c \in R$ takový, že $a = b \times c$.

Komutativní okruh R nazveme *oborem integrity*, jestliže neobsahuje dělitele nuly (tj. platí-li pro nějaké $a, b \in R$ rovnost $a \times b = 0$, pak je buď $a = 0$ nebo $b = 0$.

Komutativní okruh, ve kterém má každý nenulový prvek inverzi vzhledem k operaci $\times$, nazýváme *tělesem*.

Uveďme nyní několik příkladů právě definovaných pojmů z oblasti číselných algeber. Množina celých čísel tvoří obor integrity, který označujeme $\mathbb{Z}$. Nejběžnější příklady těles tvoří racionální čísla $\mathbb{Q}$, reálná čísla $\mathbb{R}$ a komplexní čísla $\mathbb{C}$. Všechny uvedené okruhy jsou příklady nekonečných okruhů. Jako příklad konečného okruhu může posloužit okruh zbytkových tříd modulo n (sčítání a násobení se provádí modulo n), který se značí $\mathbb{Z}_n$, případně $\mathbb{Z}/n\mathbb{Z}$. Dá se snadno ukázat, že pokud je p prvočíslo, je okruh $\mathbb{Z}_p$ konečným tělesem. Na druhé straně, je-li číslo n složené, není okruh $\mathbb{Z}_n$ nejen tělesem, není dokonce ani oborem integrity (jako příklad může posloužit rovnost $2 \times 3 = 0$, uvažovaná v okruhu $\mathbb{Z}_6$). Poznamenejme, že tato vlastnost je společná všem konečným okruhům. Platí totiž (snadné) tvrzení, že konečný obor integrity je vždy i tělesem.

3.2 Polynomy

Definice 3.2. Pro libovolný komutativní okruh R definujeme okruh polynomů $R[x]$ nad R jako množinu formálních výrazů

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0,$$

kde n je libovolné přirozené číslo a $a_i \in R$ pro $0 \leq i \leq n$. Součet polynomů $P(x) = \sum_{k=0}^{m} a_k x^k$ a $Q(x) = \sum_{k=0}^{n} b_n x^n$ definujeme předpisem

$$P(x) + Q(x) = \sum_{k=0}^{n} r_k x^k,$$

kde $c_k = a_k + b_k$ pro $0 \leq k \leq \max(m, n)$, přitom nedefinované hodnoty a_k nebo b_k uvažujeme jako nulové. Součin polynomů $P(x)$ a $Q(x)$ definujeme vztahem

$$P(x) \cdot Q(x) = \sum_{k=0}^{m+n} c_k x^k,$$

kde $c_k = \sum_{i+j=k} a_i b_j$.

Neutrálním prvkem je tzv. *nulový polynom*, který má všechny koeficienty nulové (značíme 0), jednotkovým prvkem pak polynom, který má jediný nenulový koeficient $a_0 = 1$.

Stupněm $\deg(P(x))$ nenulového polynomu $P(x) = \sum_{k=0}^{m} a_k x^k$ nazýváme největší přirozené číslo n takové, že $a_n \neq 0$. Příslušný koeficient a_n nazýváme *vedoucím koeficientem* (značíme $\mathrm{lcoeff}(P(x))$) a term $a_n x^n$ *vedoucím termem* polynomu $P(x)$ (značíme $\mathrm{lterm}(P(x))$). Poznamenejme, že z technických důvodů často definujeme stupeň nulového polynomu jako $-\infty$. Polynomy stupně 0 nazýváme *konstantními polynomy.*

Je důležité si uvědomit že okruhem R v předcházející definici může být rovněž okruh polynomů. V tomto případě mluvíme o *okruhu polynomů více proměnných* a místo $((R[x_1])[x_2] \dots)[x_n]$ píšeme $R[x_1, x_2, \dots, x_n]$ (často též pro zjednodušení $R[\boldsymbol{x}]$).

3.3 Dělitelnost v okruzích

Připomeňme, že v části 3.1 jsme definovali pojem dělitelnosti v okruhu: prvek $b \in R$ je dělitelem prvku $a \in R$ (značíme $b \mid a$), jestliže existuje $c \in R$ takový, že $a = b \times c$. Prvek a nazýváme *násobkem* prvku b.

Definice 3.3. Prvek $c \in R$ nazýváme *největším společným dělitelem (GCD)* prvků $a, b \in R$, jestliže $c \mid a$ a $c \mid b$, a platí-li pro každý společný dělitel d prvků a, b navíc $d \mid c$. [3] V dalším budeme používat označení $c = \gcd(a, b)$.

Prvek $c \in R$ nazýváme *nejmenším společným násobkem (LCM)* prvků $a, b \in R$, jestliže $a \mid c$ a $b \mid c$, a platí-li pro každý společný násobek d prvků a, b navíc $c \mid d$. Nejmenší společný násobek prvků a, b značíme $\mathrm{lcm}(a, b)$.

Definici největšího společného dělitele a nejmenšího společného násobku přirozeně rozšíříme na libovolný (konečný) počet argumentů předpisem

$$\gcd(a_1, \dots, a_n) = \gcd(\gcd(a_1, \dots, a_{n-1}), a_n),$$

resp.

$$\mathrm{lcm}(a_1, \dots, a_n) = \mathrm{lcm}(\mathrm{lcm}(a_1, \dots, a_{n-1}), a_n).$$

Prvky $a_1, \dots, a_n \in R$ nazveme *nesoudělnými*, jestliže $\gcd(a_1, \dots, a_n) = 1$.

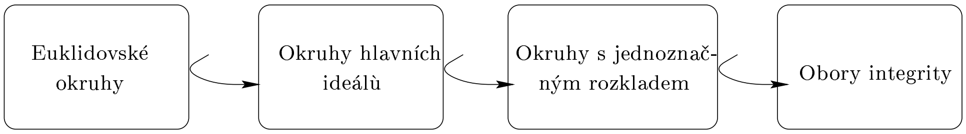
Prvky $a, b \in R$ nazveme *asociovanými*, jestliže $a \mid b$ a $b \mid a$.

Prvek $u \in R$ nazýváme *jednotkou* okruhu R, je-li asociovaný s jednotkovým prvkem okruhu R, tj. existuje-li takové $v \in R$, že $u \times v = 1$.

[3] Poznamenejme, že takováto definice GCD není jednoznačná, protože je-li prvek a největším společným dělitelem dvou prvků, je jím i libovolný jiný prvek s ním asociovaný.

Nenulový prvek p okruhu R nazýváme *ireducibilním*, jestliže nemá jiné dělitele kromě prvků asociovaných s p a jednotek.

Uveďme nyní definice speciálních podtříd okruhů, které jsou důležité pro použití v systémech počítačové algebry zejména tím, že v nich má smysl počítat největší společné dělitele, případně počítat rozklad na ireducibilní prvky. Pro tyto třídy přitom platí vztah inkluze, dá se totiž ukázat že libovolný *euklidovský okruh* je *okruhem hlavních ideálů* a libovolný *okruh hlavních ideálů* je *okruhem s jednoznačným rozkladem* na ireducibilní prvky. Tyto důkazy nejsou obtížné, nejsou však hlavním předmětem této práce, proto ukážeme pouze jednodušší vztah a zvídavého čtenáře odkážeme např. na učebnici [5].



Obrázek 3.1: Vztahy mezi jednotlivými třídami okruhů

Euklidovské okruhy

Prvním případem okruhu z hlediska dělitelnosti je tzv. *euklidovský okruh*. Je to okruh, ve kterém je možné provádět "dělení se zbytkem" tak, jak je známe například z okruhu celých čísel $\mathbb{Z}$.

Definice 3.4. Obor integrity R nazýváme *euklidovským okruhem*, jestliže existuje funkce $\lambda : R \setminus \{0\} \to \mathbb{N}$ taková, že platí

1. Pro všechna $a, b \in R \setminus \{0\}$ platí:
   jestliže $a \mid b$, pak $\lambda(a) \leq \lambda(b)$.
2. Pro všechna $a, b \in R$, kde $b \neq 0$ existují $q, r \in R$ tak, že $a = b \times q + r$, kde buď $r = 0$ nebo $\lambda(r) < \lambda(b)$.

Často také píšeme $\mathrm{quo}(a, b) = q$ a $\mathrm{rem}(a, b) = r$.

Příkladem euklidovského okruhu je např. okruh celých čísel $\mathbb{Z}$, kde má funkce λ tvar $\lambda(a) = |a|$. Jiným příkladem může být okruh polynomů nad tělesem, kde $\lambda(P(x)) = \deg(P(x))$. Na druhé straně, okruh $\mathbb{Z}[x]$ není euklidovský. Tento příklad ukazuje, že vlastnost požadovaná v definice je pěkná až příliš, protože mnoho z prakticky užitečných okruhů tuto vlastnost nesplňuje.

Okruhy hlavních ideálů

Mezistupeň mezi euklidovskými okruhy a okruhy s jednoznačným rozkladem tvoří tzv. *okruhy hlavních ideálů*. Uveďme nejprve definici pojmu *ideál*.

Definice 3.5. *Ideálem* I okruhu R nazveme takovou podmnožinu nosné množiny okruhu R, která splňuje následující axiomy:

1. Jsou-li $a, b \in I$, pak rovněž $a + b \in I$.
2. Je-li $a \in I$ a $r \in R$, pak $r \times a \in I$.

Významnou úlohu mezi ideály hrají takzvané *hlavní ideály*, což jsou ideály tvořené násobky jediného prvku (tedy I je hlavní ideál, jestliže $I = \langle a \rangle = \{r \times a \mid r \in R\}$).

Definice 3.6. Okruh R nazýváme *okruhem hlavních ideálů*, jestliže je každý ideál okruhu R hlavním ideálem.

Jak jsme již uvedli, libovolný *euklidovský okruh* je rovněž *okruhem hlavních ideálů*. Toto tvrzení dokážeme.

Buď R euklidovský okruh a buď $I \subseteq R$ ideál. Buď $a \in I \setminus \{0\}$ s nejmenší hodnotou funkce λ. Ukážeme, že $I = \langle a \rangle$. Zřejmě stačí dokázat $I \subseteq \langle a \rangle$. Buď tedy $b \in I$. Potom, protože R je euklidovský, existují $c, d \in R$ taková, že $b = c \times a + d$ a přitom $d = 0$ nebo $\lambda(d) < \lambda(a)$. Protože ale $d = b - c \times a \in I$, z podmínky minimality hodnoty $\lambda(a)$ plyne $d = 0$ a tedy $b = c \times a \in \langle a \rangle$. Celkem tedy $I = \langle a \rangle$.

Pro úplnost bychom měli uvést příklad okruhu, který je okruhem hlavních ideálů, ale není euklidovský. Tímto okruhem je např. okruh

$$\left\{ \frac{a + b\sqrt{-19}}{2} \;\middle|\; a, b \in \mathbb{Z}, a \equiv b \pmod 2 \right\}.$$

Důkaz toho, že tento okruh není euklidovský, i toho, že je okruhem hlavních ideálů, je však mimo rámec tohoto textu. Hlavní problém je v důkazu toho, že neexistuje žádná funkce λ požadovaných vlastností. Případné zájemce můžeme znovu odkázat na příslušnou literaturu (např. [1]).

Okruhy s jednoznačným rozkladem

Definice 3.7. Obor integrity R se nazývá *okruhem s jednoznačným rozkladem* (UFD), jestliže je možné psát libovolné nenulové $a \in R$ ve tvaru součinu jednotky okruhu R a konečně mnoha ireducibilních prvků okruhu R tak, že tento rozklad je jednoznačný až na asociovanost a pořadí.

Snadno se např. ukáže, že okruh $\mathbb{Z}$ celých čísel je okruhem s jednoznačným rozkladem na ireducibilní prvky (podle definice nejsou rozklady $6 = 2 \cdot 3 = (-2) \cdot (-3)$ považovány za různé). Podobně je např. okruh polynomů $\mathbb{Z}[x]$ okruhem s jednoznačným rozkladem na ireducibilní prvky. Okruh $\mathbb{C}[x, y]$ polynomů dvou proměnných s komplexními koeficienty je sice rovněž okruhem s jednoznačným rozkladem na ireducibilní prvky, není však okruhem hlavních ideálů. (Jako příklad ideálu, který není hlavní, stačí vzít ideál $I = \langle x, y\rangle$, který obsahuje všechny polynomy z $\mathbb{C}[x, y]$ s nulovým absolutním členem.)

Do úplného výčtu zbývá uvést příklad okruhu, jenž není okruhem s jednoznačným rozkladem. Jak se ukazuje, ve skutečnosti je takových okruhů mnohem více, než by bylo matematikům milé. Buď například $R = \mathbb{Z}[\sqrt{10}] = \{a + b\sqrt{10} \mid a, b \in \mathbb{Z}\}$. Pak lze poměrně snadno ukázat, že prvky $3, 1 + \sqrt{10}, 1 - \sqrt{10}$ tohoto okruhu jsou ireducibilní, navzájem neasociované a že platí $9 = 3 \cdot 3 = -(1 + \sqrt{10}) \cdot (1 - \sqrt{10})$.

3.4 Podílová tělesa

Je-li R obor integrity, lze jej jednoduchým způsobem rozšířit na těleso. Postup je analogický s postupem, jakým se rozšiřuje okruh $\mathbb{Z}$ celých čísel na těleso $\mathbb{Q}$ racionálních čísel — „konstrukce podílů“. V tomto odstavci naznačíme konstrukci, jakou lze toto učinit v obecném případě (pro detailní popis viz např. [8]).

Nechť je tedy R obor integrity. Uvažujme množinu

$$S = \{a/b \mid a \in R, b \in R \setminus \{0\}\}$$

(zápis a/b přitom chápejme pouze „syntakticky“, tj. jako jiný zápis uspořádané dvojice, který nám usnadní dívat se na dvojici jako na podíl). Na množině S uvažujme relaci $\sim$ definovanou předpisem:

$$a/b \sim c/d \iff a \times d = b \times c.$$

Tato relace je ekvivalencí na množině S. Na prvcích rozkladu podle ekvivalence $\sim$ lze definovat binární operace $+_\sim$ a $\times_\sim$ [4] pomocí reprezentantů předpisem

$$[a/b]_\sim +_\sim [c/d]_\sim = [(a \times d + b \times c)/(b \times d)]_\sim$$
$$[a/b]_\sim \times_\sim [c/d]_\sim = [(a \times c)/(b \times d)]_\sim$$

[4] Později však budou tyto operace bez nebezpečí omylu opět označovány běžnými symboly $+$ a $\times$.

Korektnost této definice se ověří snadno, stejně jako to, že faktorová množina $S/\sim$ s operacemi $+_\sim$ a $\times_\sim$ tvoří těleso. Toto těleso nazýváme *podílové těleso* oboru integrity R a značíme jej $\mathcal{Q}(R)$. Libovolný prvek a oboru integrity R je přitom zřejmě možno ztotožnit s třídou $[a/1]_\sim$.

V případě, že libovolný systém počítačové algebry implementuje práci s podílovými tělesy (což je v případě moderních systému nutnou podmínkou jeho úspěšnosti), musí být schopen vybrat z každé třídy ekvivalence tzv. *kanonického* reprezentanta (zejména kvůli určování rovnosti dvou prvků podílového tělesa). V případě, že okruh R je okruhem s jednoznačným rozkladem na ireducibilní prvky (a jak uvidíme později, existuje tedy pro každé dva prvky R jejich největší společný dělitel), je určení kanonického reprezentanta poměrně jednoduché. Například v případě tělesa racionálních čísel bývá reprezentant libovolné třídy definován jako dvojice a/b taková, že $a \in \mathbb{Z}, b \in \mathbb{N} \setminus \{0\}$ a přitom $\gcd(a, b) = 1$. [5] Je vidět, že k tomu, aby systém počítačové algebry uměl k libovolnému prvku tělesa $\mathcal{Q}(R)$ přiřadit jeho kanonického reprezentanta, je třeba výpočet největšího společného dělitele v okruhu R. Navíc, z definice operací v podílovém tělese $\mathcal{Q}(R)$ je vidět, že operace sčítání je v obecném případě náročnější než operace násobení (a to zejména v případě, že okruh R je např. okruh polynomů více proměnných, který, jak jsme ukázali, nemusí být euklidovský).

Z provedených úvah je zřejmé, že velký vliv na výkonnost konkrétního systému bude mít kvalita použitého algoritmu na výpočet největšího společného dělitele v okruhu R. V následujících odstavcích rozebereme problémy s tím spojené. Můžeme přitom naše úvahy omezit na okruhy s jednoznačným rozkladem, protože v opačném případě největší společný dělitel obecně neexistuje. Ukážeme rovněž, proč je z hlediska výpočtu největšího společného dělitele výhodné, když je okruh R euklidovským okruhem.

3.5 Největší společný dělitel

Jak jsme v předchozí části zmínili, je nutné z hlediska efektivity uvažovat při výpočtu největšího společného dělitele zvlášť okruhy, které mají vlastnost dělení se zbytkem, tj. euklidovské okruhy, a okruhy, které jsou „pouze“ okruhy s jednoznačným rozkladem. Kvůli jednoduchosti se věnujme nejprve prvnímu případu.

[5] Stejné podmínky pak musí splňovat i kanonický reprezentant v libovolné třídě obecného podílového tělesa $\mathcal{Q}(R)$, pouze podmínka, kladená na b se mění na obecnější: $b \in \mathcal{N}$, kde $\mathcal{N}$ je množina vytvořená tak, že z každé třídy rozkladu podle relace asociovanosti vybereme právě jeden prvek.

Euklidův algoritmus

V následujících odstavcích ukážeme existenci efektivního algoritmu pro výpočet největšího společného dělitele v euklidovských okruzích, tzv. *Euklidova algoritmu.* Předpokládejme, že máme dán euklidovský okruh R společně s funkcí λ splňující podmínky Definice 3.4. Základním kamenem Euklidova algoritmu je pak následující tvrzení.

Lemma 3.8. *Jsou-li $a, b \in R$, kde $b \neq 0$, a splňují-li $q, r \in R$ vztah*

$$a = b \times q + r, \text{ kde } r = 0 \text{ nebo } \lambda(r) < \lambda(b),$$

pak $\gcd(a, b) = \gcd(b, r)$.

Důkaz. Buď d libovolný společný dělitel a a b. Pak, protože $r = a - b \times q$, rovněž $d \mid r$. Je-li naopak d společným dělitelem b a r, pak je podle téže rovnosti dělitelem a. Odtud plyne požadované tvrzení. □

Jsou-li dále $a, b \in R$, kde $b \neq 0$ taková, že $\lambda(a) \geq \lambda(b)$ pak lze zavést tzv. *posloupnost zbytků* $\{r_i\}$ pro a a b následujícím způsobem:

$$r_0 = a, \quad r_1 = b, \quad r_i = \mathrm{rem}(r_{i-2}, r_{i-1})$$

pro $i \geq 2$ a dává-li předchozí rekurence smysl (tj. je-li $r_{i-1} \neq 0$).

Věta 3.9. *Je-li R euklidovský okruh, a, b, kde $b \neq 0$ jeho prvky, splňující $\lambda(a) \geq \lambda(b)$. Je-li $\{r_i\}$ posloupnost zbytků ve smyslu předchozího odstavce, pak existuje přirozené číslo k takové, že $r_{k+1} = 0$ a*

$$\gcd(a, b) = r_k.$$

Důkaz. Buď $a = r_0, b = r_1, \ldots, r_k$ příslušná posloupnost zbytků. Protože je podle definice $\lambda(r_0) \geq \lambda(r_1) > \ldots > \lambda(r_i)$ v případě, že tato r_i jsou nenulová a hodnoty funkce λ jsou přirozená čísla, dostáváme nutně, že nejdéle po $\lambda(a)$ krocích dostaneme nulový zbytek, např. r_{k+1} a podle předchozího lemmatu dostáváme

$$\gcd(a, b) = \gcd(b, r_2) = \cdots = \gcd(r_{k-1}, r_k) = r_k.$$

□

Dostáváme se tak k základní variantě Euklidova algoritmu založené na této větě:

Algoritmus 3.10. Pro zadané prvky a, b euklidovského okruhu R vrátí jejich největšího společného dělitele (a to jednoznačně až na asociovanost).

1. [Konec?] Je-li $b = 0$ vytiskni a jako výsledek a skonči.
2. [Euklidovské dělení] Polož $r \leftarrow \operatorname{rem}(a, b), a \leftarrow b, b \leftarrow r$ a jdi na 1.

Tento algoritmus je velice efektivní, je-li použit např. v případě okruhu $R = \mathbb{Z}$ (zde navíc existuje algoritmus *binární GCD*, viz např. [7, str. 322]) je kvadratické časové složitosti vzhledem k velikosti vstupu. Problém při výpočtu největšího společného dělitele pak nastává v okruzích polynomů nad nekonečnými tělesy, tj. např. nad $\mathbb{Q}, \mathbb{R}$ nebo $\mathbb{C}$. Ve dvou posledních případech není výpočet největšího společného dělitele obecně proveditelný kvůli nemožnosti reprezentovat nekonečné objekty, a tak testovat v 1.kroku algoritmu podmínku $b = 0$. Na druhé straně v případě okruhu $\mathbb{Q}[x]$ se při použití Euklidova algoritmu v této podobě dostaneme do potíží s velikostí koeficientů, což ukazuje následující příklad:

Příklad 3.11. Buďte $a(x), b(x)$ následující polynomy (tyto polynomy budeme v dalších částech této práce používat jako ilustrativní příklad vždy, pokud to bude z hlediska názornosti užitečné):

$$\begin{aligned} a(x) &= x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5 \\ b(x) &= 3x^6 + 5x^4 - 4x^2 - 9x + 21 \end{aligned}$$

Pak posloupnost zbytků, generovaná Euklidovým algoritmem, má následující tvar:

$$\begin{aligned} r_2(x) &= -\frac{5}{9}x^4 + \frac{1}{9}x^2 - \frac{1}{3}, \\ r_3(x) &= -\frac{117}{25}x^2 - 9x + \frac{411}{25}, \\ r_4(x) &= \frac{233150}{19773}x - \frac{102500}{6591}, \\ r_5(x) &= -\frac{1288744821}{543589225}. \end{aligned}$$

Dospěli jsme sice k jednotce okruhu $\mathbb{Q}[x]$, tj. dokázali jsme, že $\gcd(a, b) = 1$, vidíme ale neúměrný nárůst koeficientů jednotlivých zbytků posloupnosti.

Na příkladě těchto polynomů ukážeme výpočet největšího společného dělitele pomocí Euklidova algoritmu nad konečným tělesem. Později ukážeme výhody takového postupu, i nyní je však patrný rozdíl ve velikosti jednotlivých koeficientů.

Příklad 3.12. V případě polynomů

$$\begin{aligned} a(x) &= x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5 \\ b(x) &= 3x^6 + 5x^4 - 4x^2 - 9x + 21 \end{aligned}$$

uvažovaných jako polynomy nad $\mathbb{Z}_{13}$ (jinými slovy, koeficienty jsou podrobovány redukci modulo 13) dostáváme při Euklidově algoritmu následující posloupnost dělení:

$$x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5 \\ = (9x^2 + 7)(3x^6 + 5x^4 - 4x^2 - 9x + 21) + (11x^4 + 3x^2 + 4)$$

$$3x^6 + 5x^4 - 4x^2 - 9x + 21 = (6x^3 + 5x^2 + 6x + 5)(4x + 1) + 12$$
$$4x + 1 = (9x + 12) \cdot 12 + 0.$$

Protože libovolný nenulový konstantní polynom je jednotkou, dostáváme nesoudělnost původních polynomů. Snadnou úvahou, kterou v další kapitole rozvedeme, lze nahlédnout, že jsou-li dva celočíselné polynomy nesoudělné modulo nějaké prvočíslo a jejich vedoucí koeficienty nejsou tímto prvočíslem dělitelné,pak jsou tyto polynomy nesoudělné i nad celými čísly.

Okruhy s jednoznačným rozkladem

Mezi okruhy s jednoznačným rozkladem nás bude zajímat z hlediska systémů počítačové algebry jejich nejdůležitější případ, totiž okruhy polynomů. Jedním ze základním tvrzení je totiž fakt, že okruh polynomů nad okruhem R, který je okruhem s jednoznačným rozkladem, je rovněž UFD[6].

Jak jsme již uvedli, Euklidův algoritmus nemůže být použit pro výpočet největšího společného dělitele např. v případě okruhu polynomů více proměnných nebo v případě okruhu $\mathbb{Z}[x]$, protože se nejedná o euklidovské okruhy. Následující věta však zaručuje existenci (a jednoznačnost) největšího společného dělitele v okruhu s jednoznačným rozkladem na ireducibilní prvky, je tedy přirozené hledat algoritmus na jeho výpočet v UFD.

Věta 3.13. *Jsou-li $a, b \in R$ nenulové prvky okruhu s jednoznačným rozkladem, pak existuje $\gcd(a, b) \in R$ a to jednoznačně až na asociovanost.*

Důkaz. Mějme následující jednoznačné rozklady prvků a, b okruhu R na ireducibilní prvky, předpokládáme-li nějaké uspořádání na ireducibilních prvcích:

$$a = up_1^{e_1} p_2^{e_2} \dots p_k^{e_k} \dots, \qquad b = vp_1^{f_1} p_2^{f_2} \dots p_k^{f_k} \dots,$$

kde u, v jsou jednotky okruhu R. Označme pro každé přirozené číslo i jako g_i menší z e_i a f_i. Pak je zřejmě prvek

$$d = p_1^{g_1} p_2^{g_2} \dots p_k^{g_k} \dots$$

[6] Viz např. [7]

největším společným dělitelem prvků a a b. Jednoznačnost až na asociovanost je přitom zřejmá z definice GCD. □

Na základě důkazu Věty 3.13 je možné zkonstruovat naivní algoritmus pro výpočet největšího společného dělitele v okruhu s jednoznačným rozkladem.

Příklad 3.14. Buďte $a(x), b(x)$ polynomy v $\mathbb{Z}[x]$ dané vztahy

$$\begin{aligned} a(x) &= 30x^3 - 5x^2 + 5x - 10 \\ b(x) &= -9x^2 - 39x + 30. \end{aligned}$$

Pak faktorizace $a(x)$ a $b(x)$ nad celými čísly dává

$$\begin{aligned} a(x) &= 5(3x - 2)(2x^2 + x + 1) \\ b(x) &= -3(3x - 2)(x + 5), \end{aligned}$$

z čehož dostáváme $\gcd(a(x), b(x)) = 3x - 2$.

Podobně jako v případě okruhu celých čísel, kde je výpočet největšího společného dělitele kvadratické časové náročnosti narozdíl od rozkladu na prvočísla, který pravděpodobně není ani polynomiální, i zde existují efektivnější metody na výpočet gcd než hledání společných faktorů v rozkladu na ireducibilní prvky.

Definice 3.15. Je-li okruh R UFD, pak *primitivním polynomem* nad R nazveme polynom, jehož koeficienty jsou navzájem nesoudělné. Největší společný dělitel koeficientů polynomu $a(x) \in R[x]$ značíme $\mathrm{cont}(a(x))$. Polynom $a(x)$ lze pak jednoznačně (opět až na asociovanost) psát ve tvaru

$$a(x) = \mathrm{cont}(a(x)) \times b(x),$$

kde polynom $b(x) \in R[x]$ je primitivní (nazýváme jej *primitivní část polynomu* $a(x)$) a označuje se $\mathrm{pp}(a(x))$.

Základem algoritmu pro výpočet největšího společného dělitele polynomů je následující tvrzení, známé jako *Gaussovo lemma.*

Lemma 3.16 (Gaussovo lemma). *Součin primitivních polynomů je primitivní polynom.*

Důkaz. Mějme primitivní polynomy $a(x) = a_m x^m + \cdots + a_0$ a $b(x) = b_n x^n + \cdots + b_0$. Předpokládejme, že jejich součin $a(x) \cdot b(x)$ není primitivní, tj. že existuje ireducibilní prvek p okruhu R, který dělí všechny koeficienty polynomu $a(x) \cdot b(x)$. Podle předpokladu existují indexy i a j tak, že p nedělí

a_i ani b_j a předpokládejme, že jsou to nejmenší indexy s touto vlastností. Potom je koeficient u x_{i+j} polynomu $a(x) \cdot b(x)$ roven (koeficienty u x^k pro $k > m$ u $a(x)$ a $k > n$ u $b(x)$ jsou nulové)

$$a_0 b_{j+i} + a_1 b_{j+i-1} + \cdots + a_i b_j + a_{i+1} b_{j-1} + \cdots + a_{i+j} b_0$$

a protože jsou v tomto součtu všechny sčítance kromě členu $a_i b_j$ násobky p (podle předpokladu minimality) dostáváme, že tento koeficient není dělitelný p, což je ve sporu s původním předpokladem o dělitelnosti všech koeficientů polynomu $a(x) \cdot b(x)$ prvkem p. □

Z předchozího lemmatu vyplývá následující

Důsledek. Pro největší společný dělitel polynomů $a(x), b(x) \in R[x]$ platí

$$\gcd(a(x), b(x)) = \gcd(\mathrm{cont}(a(x)), \mathrm{cont}(b(x))) \times \gcd(\mathrm{pp}(a(x)), \mathrm{pp}(b(x))).$$

Přestože nad obecným okruhem neexistuje algoritmus dělení polynomů se zbytkem, který by splňoval vlastnosti Definice 3.4, existuje algoritmus pro dělení v okruzích polynomů nad UFD, který je pro naše účely téměř stejně vhodný. Můžeme si totiž všimnout, že při klasickém „dlouhém“ dělení polynomů dochází k dělení pouze vedoucím koeficientem dělitele. Toto můžeme vidět na následujícím příkladě.

Příklad 3.17. Buďte $a(x), b(x)$ polynomy nad $\mathbb{Q}$ stejné jako v Příkladě 3.11, tj.

$$\begin{aligned} a(x) &= x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5, \\ b(x) &= 3x^6 + 5x^4 - 4x^2 - 9x + 21. \end{aligned}$$

Pak pomocí dělení v okruhu $\mathbb{Q}[x]$ dostáváme

$$a(x) = b(x) \cdot \left(\frac{9}{27} x^2 - \frac{6}{27} \right) + \left(-\frac{15}{27} x^4 + \frac{3}{27} x^2 - \frac{9}{27} \right).$$

Tento vztah můžeme snadno přepsat do tvaru

$$27 a(x) = b(x)(9x^2 - 6) + (-15x^4 + 3x^2 - 9),$$

který je příkladem tzv. „pseudodělení“ polynomů.

Lemma 3.18 (Pseudodělení polynomů). *Nechť $R[x]$ je okruh polynomů nad okruhem R s jednoznačným rozkladem na ireducibilní prvky. Pro libovolný polynom $a(x) \in R[x]$ a libovolný nenulový $b(x) \in R[x]$ takové, že $\deg(a(x)) \geq \deg(b(x))$ existují polynomy $q(x), r(x) \in R[x]$, splňující*

$$\mathrm{lcoeff}(b(x))^{\delta+1} a(x) = b(x) \cdot q(x) + r(x) \quad a \ \deg(r(x)) < \deg(b(x)),$$

kde $\delta = \deg(a(x)) - \deg(b(x))$.

Důkaz. Jak jsme již dříve poznamenali, provádíme-li běžné dělení polynomů $a(x), b(x)$ nad podílovým tělesem $\mathcal{Q}(R)$ okruhu R, dělíme ve skutečnosti pouze prvkem $\mathrm{lcoeff}(b(x))$ a takovéto dělení provádíme nejvýše $\deg(a(x)) - \deg(b(x)) + 1$ krát. Proto je toto tvrzení přímým důsledkem toho, že okruh $\mathcal{Q}(R)[x]$ je euklidovským okruhem. □

Poznámka. Polynomy $q(x)$ a $r(x)$ z předchozího lemmatu nazýváme *pseudopodíl*, resp. *pseudozbytek* a značíme $\mathrm{pquo}(a(x), b(x))$, resp. $\mathrm{prem}(a(x), b(x))$. Podobně jako u běžného dělení jsou i zde polynom $q(x)$ a $r(x)$ určeny jednoznačně, lze je tedy počítat pomocí běžného dělení se zbytkem polynomu $\mathrm{lcoeff}(b(x))^{\delta+1}a(x)$ polynomem $b(x)$.

Lemma 3.19. *Za stejných podmínek jako v Lemmatu 3.18, požadujeme-li navíc, aby polynomy $a(x)$ a $b(x)$ byly primitivní, dostáváme*

$$\gcd(a(x), b(x)) = \gcd(b(x), \mathrm{pp}(\mathrm{prem}(a(x), b(x)))).$$

Důkaz. Za předpokladu rovnosti z Lemmatu 3.18 zřejmě dostáváme použitím stejných argumentů jako v důkazu Lemmatu 3.8

$$\gcd(\mathrm{lcoeff}(b(x))^{\delta+1} \cdot a(x), b(x)) = \gcd(b(x), r(x))$$

a použijeme-li na tuto rovnost důsledek Gaussova lemmatu, dostaneme

$$\begin{aligned}&\gcd(\mathrm{lcoeff}(b(x))^{\delta+1} \cdot a(x), b(x))\\ &\qquad = \gcd(\mathrm{lcoeff}(b(x))^{\delta+1}, 1) \cdot \gcd(a(x), b(x)) = \gcd(a(x), b(x)),\end{aligned}$$

protože $a(x)$ i $b(x)$ jsou podle předpokladů primitivní polynomy. Podobně dostaneme

$$\gcd(b(x), r(x)) = \gcd(b(x), \mathrm{pp}(r(x))).$$

Požadované tvrzení je tak dokázáno. □

Bezprostředně z předchozího lemmatu plyne následující algoritmus.

Algoritmus 3.20. Pro zadané polynomy $a(x), b(x)$ nad UFD R vrátí jejich největšího společného dělitele.

1. [Redukuj na primitivní] Vypočti $d \leftarrow \gcd(\mathrm{cont}(a(x)), \mathrm{cont}(b(x)))$. Polož $a(x) \leftarrow \mathrm{pp}(a(x)); b(x) \leftarrow \mathrm{pp}(b(x))$.

2. [Pseudodělení] Vypočítej pomocí pseudodělení $r(x) \leftarrow \mathrm{prem}(a(x), b(x))$. Je-li $r(x) = 0$, jdi na 4. Je-li $r(x)$ konstantní, polož $r(x) \leftarrow 1$ a jdi na 4.

3. [Redukuj zbytek] Polož $a(x) \leftarrow b(x)$, $b(x) \leftarrow \mathrm{pp}(r(x))$ a jdi na 2.
4. [Konec] Vytiskni $d \cdot b(x)$ a skonči.

Příklad 3.21. Uvažujme opět dvojici polynomů $a(x), b(x)$ nad celými čísly:

$$\begin{aligned} a(x) &= x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5 \\ b(x) &= 3x^6 + 5x^4 - 4x^2 - 9x + 21 \end{aligned}$$

Potom předchozí algoritmus produkuje následující posloupnost pseudodělení:

$$\begin{aligned} 27a(x) &= b(x)(9x^2 - 6) + 3(-5x^4 + x^2 - 3) \\ 125b(x) &= (5x^4 - x^2 + 3)(75x^2 + 140) + 45(-13x^2 - 25x + 49) \\ 13^3(5x^4 - x^2 + 3) &= (13x^2 + 25x - 49)(845x^2 - 1625x + 6141) \\ &\quad + 50(-4663x + 6150) \\ 4663^2(13x^2 + 25x - 49) &= (4663x - 6150)(60619x + 196525) \\ &\quad + 143193869 \cdot 1 \end{aligned}$$

Posloupnost zbytků je tedy

$$\begin{aligned} r_2(x) &= 5x^4 - x^2 + 3 \\ r_3(x) &= 13x^2 + 25x - 49 \\ r_4(x) &= 4663x - 6150 \\ r_5(x) &= 1 \end{aligned}$$

a je vidět, že oproti Příkladu 3.11 se zde vyskytují menší koeficienty u jednotlivých polynomů a navíc se veškeré výpočty odehrávají v původním okruhu koeficientů $\mathbb{Z}$.

Na druhé straně, v případě, že okruh nad kterým pracujeme, je komplexnější než okruh $\mathbb{Z}$ (např. okruh polynomů několika proměnných), významnou část svého času stráví algoritmus ve výpočtech GCD v tomto okruhu (totiž ve výpočtu primitivní části pseudozbytku v kroku 3.). Z tohoto důvodu je snaha najít metody, které budou kompromisem mezi požadavkem na co nejmenší koeficienty jednotlivých polynomů a na čas strávený redukcí zbytků.[7] Těmto metodám je věnována následující kapitola.

[7] V případě, že bychom zvolili druhou krajní možnost a neredukovali pseudozbytky vůbec, dostali bychom obecně exponenciální nárůst velikosti koeficientů u těchto zbytků. Viz např. [3, str. 282].

Kapitola 4

Algoritmy na výpočet GCD polynomů

V této kapitole ukážeme několik typů algoritmů, které lze použít pro výpočet největšího společného dělitele polynomů nad okruhem s jednoznačným rozkladem. Prvním typem algoritmů, který ukážeme v části 4.1 budou algoritmy vycházející z Euklidova algoritmu a v nichž navážeme předchozí kapitolu. Algoritmus, který podrobně rozebereme, využívá tzv. *subresultantu.*

V části 4.2 využijeme přechodu z okruhů s jednoznačným rozkladem do euklidovských okruhů pomocí vhodných homomorfismů (např. z okruhu $\mathbb{Z}[x]$ do okruhu polynomů nad konečným tělesem). Ukážeme, že ze znalosti největšího společného dělitele v těchto jednodušších okruzích dokážeme získat vhodné informace o GCD v původním okruhu. Z praktických výpočtů se ukazuje, že tento typ algoritmů je obecně výhodnější než typ předchozí. U těchto typů algoritmů nebudeme provádět podrobné rozbory korektnosti algoritmů a čtenáře odkážeme na příslušnou literaturu.

V další části této kapitoly dovedeme tento přístup ještě dál a uvedeme heuristický algoritmus, který výpočet GCD v okruhu polynomů s celočíselnými koeficienty převádí na výpočet GCD v okruhu celých čísel.

V poslední části kapitoly prezentujeme algoritmus, který je ze všech uváděných algoritmů nejlepším kompromisem mezi efektivitou a globální použitelností, algoritmus využívající tzv. *Henselova lemmatu o posunutí.* Fakticky jde o efektivní modifikaci modulárních algoritmů.

V celé této kapitole, pokud nebude řečeno jinak, bude R znamenat libovolný okruh s jednoznačným rozkladem na ireducibilní prvky (UFD).

4.1 Polynomiální zbytkové posloupnosti

Definice 4.1. Buďte $a(x), b(x) \in R[x]$ polynomy, splňující $\deg(a) \geq \deg(b)$. *Polynomiální zbytkovou posloupností (PRS)* pro A a B nazveme posloupnost $r_0(x) = a(x), r_1(x) = b(x), \ldots, r_k(x)$ polynomů z $R[x]$, pro kterou platí:

1. $\alpha_i \cdot r_{i-1}(x) = q_i(x) \cdot r_i(x) + \beta_i \cdot r_{i+1}(x)$, kde $\alpha_i, \beta_i \in R$ a $\deg(r_{i+1}) < \deg(r_i)$,

2. $\mathrm{prem}(r_{k-1}, r_k) = 0$.

Z Lemmatu 3.18 je snadno vidět, že v případě primitivních polynomů $a(x), b(x)$ je $\gcd(a(x), b(x)) = \mathrm{pp}(r_k(x))$.

Pro účely jednotného značení následujících algoritmů fixujme do konce této části označení $l_i = \mathrm{lcoeff}(r_i(x)), n_i = \deg(r_i(x)), \delta_i = n_{i-1} - n_i, m = \deg(a(x)), n = \deg(b(x))$. Již v předchozí části jsme ukázali v algoritmu 3.20 příklad tzv. *primitivní* PRS, kde $\alpha_i = l_i^{\delta_i+1}$ a $\beta_i = \mathrm{cont}(\mathrm{prem}(r_{i-1}(x), r_i(x)))$. Druhým extrémním případem, kterého jsme se dotkli, byl typ PRS, někdy nazývaný *euklidovská* posloupnost, daná parametry $\alpha_i = l_i^{\delta_i+1}$ a $\beta_i = 1$. Rovněž jsme naznačili, že ani jeden z těchto extrémů není příliš vhodný, proto se v další části budeme zabývat vhodnější volbou parametrů α_i, β_i, která bude charakterizovat různé metody pro výpočet GCD. Přitom běžnou volbou parametru α_i (jak je patrné i z Lemmatu 3.18) je $\alpha_i = l_i^{\delta_i+1}$. Cílem efektivnějších metod je tedy volit parametr β_i v závislosti na vstupních polynomech tak, aby koeficienty polynomů r_i byly pokud možno co nejmenší (extrémem je *primitivní* PRS), na druhé straně, aby byla co nejmenší cena za redukci společného dělitele koeficientů (zde je extrémním přístupem *euklidovská* PRS).

První metodou tohoto typu byla tzv. *redukovaná* PRS, navržená v polovině minulého století J. J. Sylvesterem. Spočívala v následující volbě parametrů: $\alpha_i = l_i^{\delta_i+1}, \beta_1 = 1, \beta_i = \alpha_{i-1}$ pro $2 \leq i \leq k$. Tato volba je ideální v případě, kdy posloupnost zbytků je tzv. *normální*, což znamená, že $\delta_i = 1$ pro všechna i. V tomto případě porostou koeficienty jednotlivých polynomů zhruba lineárně. Naproti tomu, v případě, že posloupnost zbytku není normální, může velikost koeficientů růst exponenciálně podobně jako u euklidovské PRS.

Příklad 4.2. Uvažujeme-li opět tytéž polynomy

$$a(x) = x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5$$
$$b(x) = 3x^6 + 5x^4 - 4x^2 - 9x + 21$$

jako v předchozích cvičeních, dostáváme redukovanou PRS

$$
\begin{aligned}
r_2(x) &= -15x^4 + 3x^2 - 9 \\
r_3(x) &= 585x^2 + 1125x - 2205 \\
r_4(x) &= -18885150x + 24907500 \\
r_5(x) &= 527933700.
\end{aligned}
$$

Tuto posloupnost nebudeme dále zkoumat, proto necháme bez odpovědi otázku, zda je definice *redukované* PRS korektní, tj. zda jsou opravdu vznikající polynomy $r_i(x)$ prvky okruhu $R[x]$. Podobnou otázku budeme řešit u následující posloupnosti, případný zájemce si tedy může detaily doplnit podle našeho příštího postupu.

Vylepšení redukované PRS navrhli nezávisle na sobě pánové G. E. Collins a W. S. Brown okolo roku 1970. Metoda využívá úzkého vztahu teorie resultantů s tímto problémem, proto se posloupnost zbytků nazývá *subresultant* PRS[1]. Metoda je dána následujícími parametry:

$$\alpha_i = l_i^{\delta_i+1}, \beta_1 = 1, \beta_i = l_{i-1} \cdot h_i^{\delta_i} \text{ pro } 2 \leq i \leq k,$$

kde

$$h_1 = 1, h_i = l_{i-1}^{\delta_{i-1}} \cdot h_{i-1}^{1-\delta_{i-1}} \text{ pro } 2 \leq i \leq k.$$

Přes své krkolomné vyjádření pomocí parametrů α_i, β_i je metoda subresultantu implementačně velmi jednoduchá, jak ukazuje následující algoritmus (tento algoritmus je doplněn o provedení úplné redukce největšího společného dělitele každých 10 dělení).

Algoritmus 4.3. K daným polynomům $a(x), b(x)$ s koeficienty v okruhu s jednoznačným rozkladem R algoritmus vypočítá jejich největší společný dělitel. Předpokládáme přitom, že máme k dispozici algoritmus pro výpočet GCD v okruhu R.

1. [Inicializace] Je-li $\deg(a(x)) < \deg(b(x))$, vyměň $a(x)$ a $b(x)$. Je-li $b(x) = 0$, vytiskni $a(x)$ a skonči, jinak polož $a(x) \leftarrow \mathrm{pp}(a(x)), b(x) \leftarrow \mathrm{pp}(b(x)), d \leftarrow \gcd(\mathrm{cont}(a(x)), \mathrm{cont}(b(x))), g \leftarrow 1, h \leftarrow 1$ a $c \leftarrow 0$.

2. [Pseudodělení] Polož $\delta \leftarrow \deg(a(x)) - \deg(b(x))$. Pomocí pseudodělení vypočítej $r(x) \leftarrow \mathrm{prem}(a(x), b(x))$. Je-li $r(x) = 0$, jdi na 4. Je-li $\deg(r(x)) = 0$, polož $b(x) \leftarrow 1$ a jdi na 4.

[1] Protože se mi nepodařilo najít dostatečně vhodný český ekvivalent, budu raději používat anglického termínu.

3. [Redukuj zbytek] Polož $a(x) \leftarrow b(x), b(x) \leftarrow r(x)/(gh^\delta)$ a $c \leftarrow c+1$. Pokud je $c < 10$, polož $g \leftarrow \mathrm{lcoeff}(a(x)), h \leftarrow h^{1-\delta}g^\delta$ a jdi na 2. Jinak polož $a(x) \leftarrow \mathrm{pp}(a(x)), b(x) \leftarrow \mathrm{pp}(b(x)), g \leftarrow 1, h \leftarrow 1$ a $c \leftarrow 0$ a jdi na 2. (Jak ukážeme později, všechna dělení v tomto kroku se provádějí pouze v rámci okruhu R.)

4. [Konec] Vytiskni $d \cdot \mathrm{pp}(b(x))$ a skonči.

Příklad 4.4. Ukažme PRS, která vznikne aplikací předešlého algoritmu na polynomy

$$\begin{aligned} a(x) &= x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5 \\ b(x) &= 3x^6 + 5x^4 - 4x^2 - 9x + 21 \end{aligned}$$

$$\begin{aligned} r_2(x) &= -15x^4 + 3x^2 - 9 \\ r_3(x) &= 65x^2 + 125x - 245 \\ r_4(x) &= -9326x + 12300 \\ r_5(x) &= 260708. \end{aligned}$$

Je vidět, že i zde, podobně jako u redukované PRS, se veškeré aritmetické operace odehrávají v původním okruhu R a rovněž velikost polynomů je omezena. Případná úplná redukce největšího společného dělitele koeficientů jednou za deset kroků je vhodným kompromisem mezi požadavkem na efektivitu a velikost koeficientů.

Dokažme nyní korektnost uvedeného algoritmu, tj. to, že koeficienty veškerých polynomů, vyskytujících se ve výpočtu, patří do původního UFD R. Uvažujme tedy PRS tvořenou polynomy $r_0(x), r_1(x), r_2(x), \ldots, r_k(x)$, kde $r_0(x) = a(x)$ a $r_1(x) = b(x)$. V označení zavedeném v úvodu této části nyní máme

$$\begin{aligned} l_1^{\delta_1+1} r_0(x) &= q_1(x) \cdot r_1(x) + l_0 h_1^{\delta_1} r_2(x), \\ l_2^{\delta_2+1} r_1(x) &= q_2(x) \cdot r_2(x) + l_1 h_2^{\delta_2} r_3(x), \\ l_3^{\delta_3+1} r_2(x) &= q_3(x) \cdot r_3(x) + l_2 h_3^{\delta_3} r_4(x), \end{aligned}$$

kde $n_2 > n_3 > n_4$ a podobně dále. Musíme ukázat, že všechny hodnoty $l_{i-1}h_i^{\delta_i}$ dělí příslušné zbytky při dělení polynomů a rovněž, že hodnoty h_i jsou prvky okruhu R. Kompletní matematický důkaz je poměrně komplikovaný (a je uveden v [3]), my se omezíme na intuitivnější důkaz, vycházející z [7].

Budeme se snažit ukázat, že polynomy $r_{j+1}(x)$ patří pro $j \geq 1$ do okruhu R. Uveďme nejprve, jak bude vypadat obecný důkaz na konkrétním příkladu pro polynomy z Příkladu 4.4. Píšeme-li $r_0(x) = a(x) = a_8x^8 + a_7x^7 + \cdots + a_0$, $r_1(x) = b(x) = b_6x^6 + b_5x^5 + \cdots + b_0$, $r_4(x) = e_1x + e_0$ a $r_5(x) = f_0$, dostáváme $h_1 = 1$, $h_2 = b_6^2$, $h_3 = c_4^2/b_6^2$ a $h_4 = d_2^2b_6^2/c_4^2$. Ukážeme například, že koeficienty polynomu $r_3(x)$ jsou v okruhu R. Uvažujme proto následující matici

$$M = \begin{array}{c} \begin{pmatrix} a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 & 0 & 0 \\ 0 & a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 & 0 \\ 0 & 0 & a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 \\ b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 & 0 \\ 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 \\ 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 \\ 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 \end{pmatrix} \end{array} \begin{array}{c} A_2 \\ A_1 \\ A_0 \\ B_4 \\ B_3 \\ B_2 \\ B_1 \\ B_0 \end{array}$$

První dvě pseudodělení uvedeného algoritmu spočívají v rovnostech

$$\begin{aligned} b_6^3r_0(x) &= r_1(x)q_1(x) + r_2(x) \\ c_4^3r_1(x) &= r_2(x)q_1(x) + b_6^5r_3(x) \end{aligned}$$

První rovnost nám prakticky umožňuje provést s maticí M určité operace. Vynásobíme řádek označený A_2 číslem b_6^3 a odečteme od něj vhodné násobky (odpovídající koeficientům polynomu $q_1(x)$) řádků označených symboly B_4, B_3 a B_2. Dostaneme tak řádek s koeficienty polynomu $r_2(x)$ v příslušných sloupcích. Tutéž operaci můžeme analogicky provést i s řádky A_1 a A_0. Dostaneme tak matici (při určité permutaci řádků)

$$M_1 = \begin{array}{c} \begin{pmatrix} b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 & 0 \\ 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 \\ 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 \\ 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 \\ 0 & 0 & 0 & 0 & c_4 & c_3 & c_2 & c_1 & c_0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & c_4 & c_3 & c_2 & c_1 & c_0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & c_4 & c_3 & c_2 & c_1 & c_0 \end{pmatrix} \end{array} \begin{array}{c} B_4 \\ B_3 \\ B_2 \\ B_1 \\ B_0 \\ C_2 \\ C_1 \\ C_0 \end{array}$$

Použitím týchž operací na druhé pseudodělení v algoritmu dostaneme matici

$$M_2 = \begin{array}{c} \begin{pmatrix} b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 & 0 \\ 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 \\ 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 \\ 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & 0 & 0 & 0 & c_4 & c_3 & c_2 & c_1 & c_0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & c_4 & c_3 & c_2 & c_1 & c_0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & c_4 & c_3 & c_2 & c_1 & c_0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & d_2 & d_1 & d_0 \end{pmatrix} \end{array} \begin{array}{c} B_4 \\ B_3 \\ B_2 \\ B_1 \\ C_2 \\ C_1 \\ C_0 \\ D_0 \end{array}$$

Označíme-li jako M', resp. M_2' libovolnou odpovídající čtvercovou podmatici matice M, resp. M_2, pak ze způsobu, jakým jsme matici M_2 odvodili z matice M dostáváme, že

$$b_6^3 \cdot b_6^3 \cdot b_6^3 \cdot (c_4^3/b_6^5) \cdot \det M' = \pm \det M_2'.$$

Vybereme-li například z obou matic sedm prvních sloupců a poslední sloupec, dostaneme rovnost

$$b_6^3 \cdot b_6^3 \cdot b_6^3 \cdot (c_4^3/b_6^5) \cdot \det \begin{pmatrix} a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & 0 \\ 0 & a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & 0 \\ 0 & 0 & a_8 & a_7 & a_6 & a_5 & a_4 & a_0 \\ b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & 0 \\ 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & 0 \\ 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & 0 \\ 0 & 0 & 0 & 0 & b_6 & b_5 & b_4 & b_0 \end{pmatrix} = \pm b_6^4 \cdot c_4^3 \cdot d_0.$$

Z toho dostáváme, že d_0 patří do okruhu R. Podobně se příslušnost do okruhu R ukáže i pro d_0 a d_1, pouze je nutné zvolit jinou čtvercovou podmatici.

V obecném případě se ukáže naprosto analogicky tvrzení, že koeficienty polynomu $r_{j+1}(x)$ patří do R pro $j \geq 1$. Začneme opět s maticí M složenou ze dvou schodovitých bloků: první tvoří řádky označené jako $A_{n_1-n_j}, \ldots, A_0$ a druhý řádky $B_{n_0-n_j}, \ldots, B_0$. Naznačenými operacemi nad touto maticí, určenými posloupností příslušných pseudodělení, dostaneme matici M_{j-1}, ssetavenou z řádků $B_{n_0-n_j}, \ldots, B_{n_2-n_j}, C_{n_1-n_j}, \ldots, C_{n_3-n_j}, \ldots, Q_{n_{j-1}-n_j}, \ldots, Q_0$ a R_0, kde řádek R_0 je tvořen koeficienty polynomu $r_{j+1}(x)$. Tak jako v ukázkovém případě platí, že jsou-li M' a M_{j-1}' odpovídající čtvercové podmatice matic M a M_{j-1}, dostaneme

$$\left(l_1^{\delta_1+1}/l_0 h_1^{\delta_1}\right)^{n_1-n_j+1} \ldots \left(l_j^{\delta_{j-1}+1}/l_{j-1} h_{j-1}^{\delta_{j-1}}\right)^{n_j-n_j+1} \times \det M' = \pm l_1^{n_0-n_2} l_2^{n_1-n_3} \ldots l_j^{n_{j-1}-n_j+1} \cdot r_t,$$

kde r_t je příslušný koeficient polynomu $r_{j+1}(x)$, odpovídající vybranému sloupci v submatici M'_{j-1}. Výpočtem s využitím definice koeficientů h_i se poměrně snadno ukáže, že uvedená rovnost se podobně jako v ukázkovém případě redukuje na

$$\det M' = \pm r_t.$$

Protože je matice M' sestavena z koeficientů polynomů $a(x)$ a $b(x)$, je zřejmě $\det M' \in R$ a tedy $r_t \in R$.

Podobným způsobem se ukáže, že všechna h_i patří do okruhu R a že tedy všechny operace uvedeného algoritmu se skutečně odehrávají v okruhu R bez nutnosti práce v podílovém tělese. Ukážeme opět na našem příkladě, že $h_3 \in R$. Uvažujme matici

$$M = \left(\begin{array}{cccccccccc} a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 & 0 \\ 0 & a_8 & a_7 & a_6 & a_5 & a_4 & a_3 & a_2 & a_1 & a_0 \\ b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 \\ 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 \\ 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 \end{array}\right) \begin{array}{c} A_1 \\ A_0 \\ B_3 \\ B_2 \\ B_1 \\ B_0 \end{array}$$

Provedeme-li operace podle prvního pseudodělení, dostaneme matici

$$M_1 = \left(\begin{array}{cccccccccc} b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 & 0 \\ 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 & 0 \\ 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & 0 & 0 & b_6 & b_5 & b_4 & b_3 & b_2 & b_1 & b_0 \\ 0 & 0 & 0 & 0 & c_4 & c_3 & c_2 & c_1 & c_0 & 0 \\ 0 & 0 & 0 & 0 & 0 & c_4 & c_3 & c_2 & c_1 & c_0 \end{array}\right) \begin{array}{c} B_3 \\ B_2 \\ B_1 \\ B_0 \\ C_1 \\ C_0 \end{array}$$

a vybereme-li prvních šest sloupců těchto matic (označíme M', resp. M'_1), dostaneme

$$b_6^3 \cdot b_6^3 \cdot \det M' = \pm \det M'_1,$$

což dává $\det M' = \pm c_4^2/b_6^2 = \pm h_3$ a protože matice M byla opět tvořena koeficienty polynomů $a(x)$ a $b(x)$, je $h_3 \in R$. Diskuse obecného případu je podobným rozšířením tohoto příkladu jako zobecnění předchozí, proto se ji odvážíme vynechat.

4.2 Modulární metody

V předchozí části jsme ukázali několik metod, kterými se lze bránit růstu velikosti koeficientů jednotlivých polynomů, vznikajících v průběhu výpočtu největšího společného dělitele polynomů nad okruhem, který není tělesem a není tedy možné použít některou variantu Euklidova algoritmu. V této části ukážeme, jak lze s výhodou využít přenesení problému do jednoduššího oboru, přestože při tomto přenosu můžeme ztratit část informace. Zpravidla přitom půjde o přenos z původního okruhu polynomů, který je UFD (a má tedy smysl v něm definovat největšího společného dělitele) do euklidovského okruhu, kde je možné použít Euklidův algoritmus. Přes již zmíněnou ztrátu informace, může výpočet GCD v jediném takovém „jednodušším“ okruhu přinést dostatek informace k určení GCD v původním okruhu, což ukazuje následující lemma.

Lemma 4.5. *Buďte $R[x]$ a $R'[x]$ okruhy polynomů s jednoznačným rozkladem a $f : R[x] \to R'[x]$ homomorfismus těchto okruhů, přirozeně indukovaný homomorfismem okruhů R a R'. Jsou-li nyní $a(x), b(x) \in R[x]$ polynomy a $c(x) = \gcd(a(x), b(x))$ takový, že $f(\mathrm{lcoeff}(c(x))) \neq 0$, pak*

$$\deg(\gcd(f(a(x)), f(b(x)))) \geq \deg(\gcd(a(x), b(x))).$$

Důkaz. Buďte $d(x), e(x) \in R[x]$ polynomy splňující $a(x) = c(x) \cdot d(x)$, $b(x) = c(x) \cdot e(x)$. Potom z vlastností homomorfismů plyne, že $f(a(x)) = f(c(x)) \cdot f(d(x))$ a $f(b(x)) = f(c(x)) \cdot f(e(x))$, a tedy že $f(c(x))$ je společným dělitelem polynomů $f(a(x)), f(b(x)) \in R'[x]$. Proto je $f(c(x))$ dělitelem jejich největšího společného dělitele a má tedy nejvýše stejný stupeň. Tvrzení nyní plyne z předpokladu $f(\mathrm{lcoeff}(c(x))) \neq 0$. ☐

Jako důsledek tohoto tvrzení uveďme jednoduchou podmínku, která se dříve často používala jako efektivní heuristika k testování ireducibility polynomů.

Důsledek. Za stejných podmínek jako v předešlém lemmatu, pokud navíc předpokládáme, že polynomy $f(a(x))$ a $f(b(x))$ jsou ireducibilní, dostáváme, že rovněž polynomy $a(x)$ a $b(x)$ jsou ireducibilní.

Užitečnost takovéto heuristiky oceníme zejména tehdy, uvědomíme-li si, že v nejčastějším případě, ve kterém používáme uváděné algoritmy, totiž v okruzích polynomů s koeficienty z okruhu celých čísel, jsou dva náhodné polynomy soudělné takřka s nulovou pravděpodobností. Hlavní věta teorie resultantů totiž tvrdí, že k tomu aby měly dva polynomy netriviální společný faktor, je nutné (a stačí), aby byl jejich resultant nulový. Přitom resultant je (podobně jako subresultant) determinantem matice, která je sestavena z jistých koeficientů těchto polynomů. Podobným argumentem může být výpočet, provedený v [7], kde se ukazuje, že dva normované polynomy téhož stupně v rovnoměrné distribuci jsou nesoudělné modulo p s pravděpodobností $1 - 1/p$.

Příklad 4.6. Aplikujme uvedený postup na polynomy

$$
\begin{aligned}
a(x) &= x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5 \\
b(x) &= 3x^6 + 5x^4 - 4x^2 - 9x + 21
\end{aligned}
$$

Pak použitím redukce do okruhu polynomů $\mathbb{Z}_{13}[x]$ dostaneme (viz Příklad 3.12), že polynomy $a(x)$ a $b(x)$ jsou nesoudělné modulo 13 a tedy podle předchozího důsledku jsou nesoudělné jako polynomy okruhu $\mathbb{Z}[x]$.

V této části ukážeme jednoduchý algoritmus, který počítá největší společný dělitel polynomů z okruhu $\mathbb{Z}[x]$, tj. polynomů jedné proměnné s koeficienty z okruhu celých čísel. Tento algoritmus lze snadno převést na případ okruhu polynomů jedné proměnné s koeficienty z libovolného euklidovského okruhu (takový algoritmus je např. součástí systému Axiom). O tom, jak přenést uvedené výsledky na okruhy polynomů více proměnných, se zmíníme v závěru této části.

V případě okruhů s celočíselnými koeficienty se jako cílový okruh „zjednodušujícího" homomorfismu nabízí okruh polynomů nad konečným tělesem, tj. okruh $\mathbb{Z}_p[x]$, kde p je prvočíslo. Z toho, že homomorfismus f je v tomto případě počítaní s polynomy modulo p, plyne i název modulární metody. Další výhodou toho, že jde o konečné těleso, je omezenost koeficientů a v případě, že je prvočíslo p dostatečně malé na to, aby se veškeré výpočty mohly odehrávat v jednoduché přesnosti, jde o významnou přednost před předchozími algoritmy, které byly nuceny používat implementaci „dlouhých" čísel.

Základní myšlenkou této metody je výpočet největšího společného dělitele polynomů $a(x)$ a $b(x)$ modulo několik prvočísel a ze znalosti těchto výsledků určit $\gcd(a(x), b(x))$. V tomto postupu hraje klíčovou úlohu tzv. *Čínská zbytková věta*, která říká (ve své nejjednodušší formě), že známe-li zbytky čísla a po dělení různými prvočísly $p_1, \dots, p_n$, pak lze z těchto zbytků jednoznačně získat zbytek po dělení čísla a číslem $p_1 \cdots p_n$. Navíc existují efektivní algoritmy, počítající tento zbytek (viz např. Garnerův CRA algoritmus v [3], využívající zápisu celých čísel v tzv. polyadické číselné soustavě). My budeme v dalším předpokládat, že algoritmus CRA máme k dispozici (podrobnosti viz např. [2, 3]).

Může se ovšem stát, že zvolíme prvočíslo p tak nešťastně, že největší společný dělitel modulo p má mizivou vypovídací hodnotu o hledaném GCD. To ukazuje například následující příklad.

Příklad 4.7. Mějme opět celočíselné polynomy

$$
\begin{aligned}
a(x) &= x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5 \\
b(x) &= 3x^6 + 5x^4 - 4x^2 - 9x + 21
\end{aligned}
$$

a počítejme jejich největší společný dělitel modulo 2. Výsledkem aplikace Euklidova algoritmu je polynom $x + 1$, přestože původní polynomy jsou nesoudělné v okruhu $\mathbb{Z}[x]$.

Uvedený příklad ilustroval možnost, že prvočíslo p je tzv. *nešťastné*. V tom případě je stupeň GCD modulo p příliš veliký. Naštěstí je počet *nešťastných* prvočísel poměrně malý, neboť se ukazuje, že prvočíslo p může být nešťastné pouze tehdy, pokud p dělí vedoucí koeficient některého z polynomů, které vzniknou jako zbytky při PRS (např. jednotlivé polynomy $r(x)$ v Algoritmu 3.20). Podobně se v [3] ukazuje hranice pro počet takových prvočísel. Toto tvrzení uvedeme bez důkazu [2].

Lemma 4.8. *Počet nešťastných prvočísel, větších než pevně zvolené číslo N je shora omezen číslem*

$$m \cdot \log_N(2mc^2t^2),$$

kde $c = \max(\|a(x)\|, \|b(x)\|)$, $m = 1/2 \max(\deg(a(x)), \deg(b(x)))$ *a* $t = \max(t(a(x)), t(b(x)))$, *přitom* $t(p(x))$ *je počet členů polynomu* $p(x)$.

Uvedené lemma tak potvrzuje názor intuice v tom, že je dobré volit prvočíslo p co největší, ale tak, abychom pořád operovali s aritmetikou v jednoduché přesnosti.

Základním pozorováním, které ukazuje užitečnost modulárních metod je fakt, že pokud prvočíslo p není nešťastné, pak platí $f(\gcd(a(x), b(x))) = l \cdot \gcd(f(a(x)), f(b(x)))$ pro nějakou konstantu l tělesa $\mathbb{Z}_p$. Pokud bychom dokázali zjistit konstantu l, znali bychom $\gcd(a(x), b(x))$ modulo p. Zjišťování této konstanty se však v praxi vyhneme následujícím způsobem. Nejprve můžeme předpokládat, že původní polynomy $a(x), b(x)$ jsou již primitivní. Vedoucí koeficient polynomu $c(x) = \gcd(a(x), b(x))$ zřejmě dělí číslo $g = \gcd(\mathrm{lcoeff}(a(x)), \mathrm{lcoeff}(b(x)))$, proto můžeme psát $g = u \cdot \mathrm{lcoeff}(c(x))$ pro vhodnou konstantu u. Pokud bychom normovali polynomy $f(a(x))$ a $f(b(x))$ tak, aby jejich vedoucí koeficienty byly rovny $f(g)$, dostali bychom $l = u$. Pomocí CRA bychom pak místo polynomu $c(x)$ dostali $l \cdot c(x)$, ale z toho lze snadno určit polynom $c(x)$ díky Gaussovu lemmatu tím, že musí být primitivní.

Uveďme nyní tedy princip modulárního algoritmu na výpočet největšího společného dělitele.

- Vypočteme počet prvočísel, modulo která budeme počítat GCD.
- Určíme vhodné prvočíslo p a vypočteme GCD modulo p.

[2] V monografii [3] je věta uváděna pro případ polynomů více proměnných, my se omezíme na zde probíraný případ polynomů jedné proměnné.

- V případě, že prvočíslo nebylo nešťastné, začleníme pomocí CRA získaný GCD modulo p do hledaného GCD.
- Pokud bylo již prvočísel dost, vytiskneme GCD, jinak postup opakujeme s volbou nového prvočísla.

Nyní je nutné specifikovat některé kroky tohoto schématu. Především tedy, kolik prvočísel budeme potřebovat, jak poznáme, že jde o nešťastné prvočíslo a jak začlenit získané výsledky do hledaného GCD.

Pro zjištění počtu potřebných prvočísel můžeme použít horní odhad velikosti koeficientů výsledného GCD, uvedený ve [3]. Ukazuje se zde totiž, že je-li $c(x) = \gcd(a(x), b(x))$ a $\nu = \min(\deg(a(x)), \deg(b(x)))$, pak

$$\|c(x)\| \leq 2^{\nu} \cdot \min(\|a(x)\|, \|b(x)\|) \cdot \gcd(\mathrm{lcoeff}(a(x)), \mathrm{lcoeff}(b(x))).$$

To, zdali je prvočíslo nešťastné, poznáme, jak jsme již uvedli, především podle toho, že výsledný GCD modulo toto prvočíslo má příliš vysoký stupeň. Je-li tedy toto prvočíslo p několikátým v řadě a GCD modulo p má vyšší stupeň než dosud zkonstruovaná aproximace, pak jej rovnou z dalších úvah vyloučíme. Užitečným testem, který mnohá nešťastná prvočísla odhalí ještě *před* výpočtem GCD, je test na dělitelnost vedoucích koeficientů polynomů $a(x)$ a $b(x)$ tímto prvočíslem.

V případě, že jsme zjistili, že nejde o nešťastné prvočíslo, potřebujeme začlenit získaný GCD do hledaného největšího společného dělitele v $\mathbb{Z}[x]$. To se děje pomocí CRA. Po použití prvočísel $p_1, \ldots, p_k$ známe koeficienty výsledného GCD modulo $p_1 \cdots p_k$. Po aplikaci algoritmu s novým prvočíslem p můžeme snadno zjistit použitím CRA jednotlivé koeficienty modulo $p_1 \cdots p_l \cdot p$. Opakujeme tento postup tak dlouho, až zjistíme, že $p_1 \cdots p_k$ je větší než vypočtená horní hranice velikostí koeficientů výsledného GCD. Alternativním způsobem ukončení algoritmu je test dělitelnosti. V případě, že dosud zkonstruovaná „aproximace“ největšího společného dělitele $c'(x)$ dělí oba polynomy $a(x), b(x)$ v okruhu $\mathbb{Z}[x]$, je zřejmé, že platí $c'(x) \mid \gcd(a(x), b(x))$ a podle Lemmatu 4.5 dostáváme, že stupeň $c'(x)$ není menší než stupeň $c(x)$, a proto je $c'(x)$ největším společným dělitelem $a(x)$ a $b(x)$.

Algoritmus 4.9. Algoritmus MGCD vypočítá pro dané polynomy $a(x), b(x)$ jedné proměnné s celočíselnými koeficienty jejich největšího společného dělitele použitím modulárních metod.

1. [Redukuj a vypočítej hranici] Vypočítej nejdříve užitečné konstanty $\mathit{conta} \leftarrow \mathrm{cont}(a(x))$, $\mathit{contb} \leftarrow \mathrm{cont}(b(x))$, $a(x) \leftarrow \mathrm{pp}(a(x))$, $b(x) \leftarrow \mathrm{pp}(b(x))$, $\mathit{gcont} \leftarrow \gcd(\mathit{conta}, \mathit{contb})$ a dále jako g označme největší společný dělitel vedoucích koeficientů polynomů $a(x)$ a $b(x)$. Pro účely

výpočtu horní hranice velikosti koeficientů polož $primes\_product \leftarrow 1$, $candidate(x) \leftarrow 0$, $n \leftarrow \min(\deg(a(x)), \deg(b(x)))$ a $limit \leftarrow 2^n \cdot |g| \cdot \min(\|a(x)\|, \|b(x)\|)$.

2. [Nové prvočíslo] Vyber vhodně velké prvočíslo p nedělící g. Buďte dále polynomy v $\mathbb{Z}_p[x]$: $a_p(x) \leftarrow a(x) \bmod p$, $b_p(x) \leftarrow b(x) \bmod p$, $c_p(x) \leftarrow \gcd(a_p(x), b_p(x))$ a $m \leftarrow \deg(c_p(x))$. Normalizuj $c_p(x)$ tak, aby měl vedoucí koeficient kongruentní s g modulo p: $g_p \leftarrow g \bmod p$, $c_p(x) \leftarrow g_p \cdot \mathrm{lcoeff}(c_p(x))^{-1} \cdot c_p(x)$.

3. [Nešťastné p?] Je-li $m < n$, polož $primes\_product \leftarrow p$, $candidate(x) \leftarrow c_p(x)$ a $n \leftarrow m$. Pokud je $m = n$, pak použij na všechny koeficienty c_i polynomu $candidate(x)$ Čínský zbytkový algoritmus a polož $c_i \leftarrow \mathrm{CRA}(\mathrm{mod}(c_i, primes\_product), \mathrm{mod}(c_p(x)_i, p))$ a součin dosud uvažovaných prvočísel $primes\_product$ vynásob prvočíslem p.

4. [Konec?] Je-li $primes\_product \geq limit$, pak pokud $\mathrm{pp}(candidate(x))$ dělí $a(x)$ i $b(x)$, vrať $gcont \cdot candidate(x)$ a skonči. Jinak jdi na 2.

Ukažme uvedený algoritmus na konkrétním příkladu:

Příklad 4.10. Mějme $a(x), b(x) \in \mathbb{Z}[x]$ následující polynomy:

$$a(x) = 15x^4 + 103x^3 + 157x^2 - 171x - 360$$
$$b(x) = 6x^5 + 31x^4 + 47x^3 + 26x^2 + x - 15.$$

Spočítejme nejprve horní hranici koeficientů největšího společného dělitele: $limit = 2^4 \cdot 3 \cdot 47 = 2256$. Přestože by bylo výhodné zvolit jako prvočíslo p číslo 2267, kvůli prezentaci algoritmu raději zvolíme malá prvočísla. Po redukci modulo 5 dostáváme

$$a_5(x) = 3x^3 + 2x^2 + 4x, \quad b_5(x) = x^5 + x^4 + 2x^3 + x^2 + x.$$

Aplikací Euklidova algoritmu v okruhu $\mathbb{Z}_5[x]$ dostaneme $\gcd(a_5(x), b_5(x)) = x^2 + 3x$. Protože největší společný dělitel vedoucích koeficientů původních polynomů je $g = 3$, musíme nalezený GCD normovat a dostaneme $c_5(x) = 3x^2 + 4x$. Jako další prvočíslo zvolíme číslo 7. Po redukci dostáváme

$$a_7(x) = x^4 + 5x^3 + 3x^2 + 4x + 4, \quad b_7(x) = 6x^5 + 3x^4 + 5x^3 + 5x^2 + x + 6.$$

a tedy $\gcd(a_7(x), b_7(x)) = 6x^2 + 2$ a po normalizaci $c_7(x) = 3x^2 + 1$. Pomocí Čínského zbytkového algoritmu nyní dostaneme $candidate(x) = 3x^2 + 14x + 15$ a přitom je $primes\_product = 35$.

Dále redukujme modulo 11 a dostaneme

$$a_{11}(x) = 4x^4 + 4x^3 + 3x^2 + 5x + 3, \quad b_{11}(x) = 6x^5 + 9x^4 + 3x^3 + 4x^2 + x + 7.$$

Výpočet GCD dává $\gcd(a_{11}(x), b_{11}(x)) = 3x^2 + 3x + 4$. Zde není nutná redukce a je vidět, že po provedení CRA se polynom $candidate(x)$ nezmění. To je signálem toho, že jsme již nalezli správného kandidáta, přestože ještě stále není *primes_product* $>$ *limit*. V případě, že zvolíme další prvočíslo, dostaneme již definitivně

$$candidate(x) = 3x^2 + 14x + 15.$$

Testem na dělitelnost polynomů $a(x)$ a $b(x)$ nalezeným polynomem zjistíme, že jde skutečně o největšího společného dělitele těchto polynomů.

V předchozím příkladě jsme viděli, že uvedený horní odhad nemusí být vždy dostatečně dobrý, proto se nabízí některá z následujících modifikací algoritmu modulárního GCD:

- Po každé změně polynomu $candidate(x)$ vyzkoušet, zda dělí polynomy $a(x)$ a $b(x)$, pokud ano, jde o jejich největší společný dělitel.
- Zkoušet, zda polynom $candidate(x)$ dělí polynomy $a(x)$ a $b(x)$ pouze v případě, že jeho koeficienty zůstaly v tomto kroku nezměněny a je tedy naděje, že jde skutečně o největší společný dělitel $a(x)$ a $b(x)$.

Ukažme ještě příklad, ve kterém vystupuje „nešťastné“ prvočíslo.

Příklad 4.11. Mějme následující polynomy

$$\begin{aligned} a(x) &= x^4 + 25x^3 + 145x^2 - 171x - 360 \\ b(x) &= x^5 + 14x^4 + 15x^3 - x^2 - 14x - 15. \end{aligned}$$

Budeme-li redukovat modulo 5, dostaneme

$$a_5(x) = x^4 - 4, \quad b_5(x) = x^5 - x^4 - x^2 + x,$$

což dává $\gcd(a_5(x), b_5(x)) = x^4 - 4$. Přitom po redukci modulo 7 a výpočtu GCD dostaneme $\gcd(a_7(x), b_7(x)) = x^2 + 1$. Z toho vyplývá, že prvočíslo 5 bylo nešťastným a největší společný dělitel vypočítaný modulo 5 zavrhneme. Provedením algoritmu pro další prvočísla posléze zjistíme, že

$$\gcd(a(x), b(x)) = x^2 + 14x + 15.$$

Analogický postup lze zvolit pro polynomy více proměnných nad UFD. Roli jednotlivých prvočísel v předchozím algoritmu (resp. jednotlivých modulárních homomorfismů) zde hrají polynomy $x_i - a_i$ (zde výpočet polynomu modulo $x_i - a_i$ znamená vyhodnocení polynomu při substituci $x_i = a_i$). Používá se pak v podstatě stejná Čínská zbytková věta, které se však v kontextu polynomů obvykle říká *interpolace*. Obvykle je výhodné za a_i brát číslo 0, a to zejména v případě, že používáme tzv. řídkou reprezentaci polynomů.

Problémem, kterého jsme se dotkli v závěru předchozího odstavce, může být množství nutných netriviálních výpočtů, které je nutno provést v případě, že počet proměnných je velmi vysoký. V tomto případě však bývá v praxi počet nenulových koeficientů mnohem menší než v průměrném teoretickém případě, proto je zde extrémně výhodná řídká reprezentace polynomu a s tím související modifikace modulárního GCD algoritmu, kterou zde načrtneme, tzv. *řídký modulární GCD algoritmus*. V případě běžného modulárního algoritmu (v této části jej navíc ozdobíme přívlastkem *hustý*) předpokládáme, že polynom stupně n má všechny koeficienty neznámé, proto je třeba $n+1$ vyhodnocovací bodů k tomu, aby bylo interpolací možno zjistit výsledek. *Řídký* algoritmus interpolace předpokládá, že je pouze velmi málo koeficientů (oproti stupni polynomu), které jsou nenulové a je proto nutné volit podstatně méně vyhodnocovacích bodů. Tento algoritmus střídavě používá *hustou* interpolaci, která nám poskytne kandidáty na nulové koeficienty a řídkou interpolaci, která předpokládá znalost nulových koeficientů.

Na ukázku, kvůli lepšímu pochopení této metody ukažme příklad převzatý z učebnice [3]. Zároveň tak budeme demonstrovat postup popsaný pro výpočet GCD polynomů více proměnných modulárním algoritmem.

Příklad 4.12. Buďte $a(x,y,z)$ a $b(x,y,z) \in \mathbb{Z}[x,y,z]$ následující polynomy:

$$\begin{aligned} a(x,y,z) &= x^5 + 2yzx^4 + (13yz^2 - 21y^3z + 3)x^3 + (26y^2z^3 - 42y^4z^2 + 2)x^2 \\ &\quad + (39yz^2 - 63y^3z + 4yz)x + 6, \\ b(x,y,z) &= x^6 + (13yz^2 - 21y^3z + z + y)x^4 + 3x^3 \\ &\quad +(13yz^3 + 13y^2z^2 - 21y^3z^2 - 21y^4z)x^2 \\ &\quad +(13yz^2 - 21y^3z + 21 + 2y)x + 2. \end{aligned}$$

Ukážeme, jak vypočítat koeficient u x^1 v $\gcd(a(x), b(x))$. Určíme GCD modulo 11 a 17. Všimněme si, že $\deg_y(a(x)) = 4$ a $\deg_z(a(x)) = 3$, proto i GCD musí mít stupeň vzhledem k proměnné y nejvýše 4 a vzhledem k z nejvýše 3.

Vypočítejme nejprve největší společný dělitel modulo 11, tj. v okruhu polynomů $\mathbb{Z}_{11}[x,y,z]$. Prvním krokem bude, že určíme obraz tohoto GCD v okruhu $\mathbb{Z}_{11}[x,y]$ při vyhodnocení v bodě $z = 2$. Pomocí výpočtu GCD v okruhu $\mathbb{Z}_{11}[x]$ pro hodnoty $y = 1, 3, 5, -4, 4$ dostaneme

$$\begin{aligned}
\gcd(a(x,1,2), b(x,1,2)) &= x^3 - x + 2 \pmod{11} \\
\gcd(a(x,3,2), b(x,3,2)) &= x^3 + x + 2 \pmod{11} \\
\gcd(a(x,5,2), b(x,5,2)) &= x^3 + 4x + 2 \pmod{11} \\
\gcd(a(x,-4,2), b(x,-4,2)) &= x^3 + 5x + 2 \pmod{11} \\
\gcd(a(x,4,2), b(x,4,2)) &= x^3 - 5x + 2 \pmod{11}.
\end{aligned}$$

Pomocí (husté) interpolace dostaneme hledaný obraz v $\mathbb{Z}_{11}[x, y]$ jako

$$x^3 + (-3y + 2y^3)x + 2.$$

V případě běžného modulárního algoritmu pro polynomy více proměnných bychom nyní opakovali tentýž postup pro jinou náhodně zvolenou hodnotu z, místo toho nyní budeme předpokládat, že hledaný největší společný dělitel bude tvaru

$$x^3 + (ay + by^3)x + 2.$$

Nyní vypočteme pro hodnotu $z = -5$ obraz GCD v okruhu $\mathbb{Z}_{11}[x, y]$ tak, že metodou neurčitých koeficientů zjistíme hodnoty a a b pro substituci $z = -5$.

$$\begin{aligned}
\gcd(a(x,-3,-5), b(x,-3,-5)) &= x^3 - 4x + 2 \pmod{11} \\
\gcd(a(x,2,-5), b(x,2,-5)) &= x^3 + 5x + 2 \pmod{11},
\end{aligned}$$

což dává (všechny výpočty se samozřejmě provádějí modulo 11) obraz hledaného GCD modulo 11 a při substituci $z = -5$ jako

$$x^3 + (-5y - 5y^3)x + 2.$$

Analogické výpočty pro substituce $z = -3$ a $z = 5$ dávají

$$\begin{gathered}
x^3 + (-4y - 3y^3)x + 2 \\
x^3 + (-5y + 5y^3)x + 2,
\end{gathered}$$

vše modulo 11. Aplikací (husté) interpolace dostaneme kandidáta na obraz hledaného GCD modulo 11 jako

$$x^3 + (2yz^2 + y^3 z)x + 2.$$

Analogickým postupem (s využitím známého výsledku modulo 11) dostaneme jako kandidáta pro obraz GCD modulo 17 jako

$$x^3 + (-4yz^2 - 4y^3 z)x + 2,$$

což pomocí aplikace Čínské zbytkové věty na jednotlivé koeficienty dává GCD v $\mathbb{Z}[x, y, z]$ jako

$$x^3 + (13yz^2 - 21y^3z)x + 2.$$

Projdeme-li uvedený příklad podrobně, zjistíme, že pro výpočet GCD touto metodou je zapotřebí 13 výpočtů největšího společného dělitele polynomů jedné proměnné nad konečnými tělesy, zatímco v případě modulárního algoritmu je to přibližně 40 výpočtů. Jde samozřejmě o pravděpodobnostní algoritmus, takže i zde je důležitá vhodná volba prvočísel p, modulo která provádíme výpočty — čím větší jsou zvolena prvočísla, tím menší je pravděpodobnost, že náhodně zvolený polynom bude mít (nenulový) koeficient u některého členu dělitelný tímto prvočíslem. Podobně se dá pravděpodobnost výpočtu špatného kandidáta snížit na libovolně malou hodnotu volbou dostatečně velkých vyhodnocovacích bodů. Podotkněme, že se v tomto případě díky množství nezávislých výpočtů nabízí implementace pomocí paralelních výpočtů.

4.3 Heuristický algoritmus

Jak bylo řečeno dříve, úspěšnost modulárních metod vyplývá zejména z toho, že narozdíl od přirozených čísel jsou polynomy s vysokou pravděpodobností nesoudělné (např. dva nezávislé normované polynomy stupně n, rovnoměrně distribuované, jsou s pravděpodobností $1 - 1/p$ nesoudělné modulo p, další viz [7]). Proto bylo často užitečné jako jednoduchý test zkusit, zda nejsou polynomy $a(x), b(x)$ ireducibilní modulo některé prvočíslo (a tento test provést pro několik prvočísel). Pokud ano, pak podle Lemmatu 4.5 víme, že jsou nutně ireducibilní i polynomy nad původním okruhem. V této části ukážeme algoritmus, který pomocí modulárních metod spočítá největšího společného dělitele i v případě, že výchozí polynomy nejsou ireducibilní.

Heuristický algoritmus je založen na podobném principu jako modulární metody. Jsou-li $a(x), b(x)$ například polynomy jedné proměnné nad okruhem $\mathbb{Z}$ celých čísel, pak se můžeme domnívat, že $\gcd(a(\xi), b(\xi))$ je nějak svázán s $\gcd(a(x), b(x))$, kde $\xi \in \mathbb{Z}$ je vhodné číslo. Dá se dokonce předpokládat, že by bylo možné ze znalosti $\gcd(a(\xi), b(\xi))$ zkonstruovat největšího společného dělitele polynomů $a(x)$ a $b(x)$. Tento předpoklad se ukazuje pravdivým při vhodné volbě čísla ξ. Ilustrujme tuto myšlenku na jednoduchém příkladě.

Příklad 4.13. Buďte $a(x), b(x)$ naše standardní polynomy, totiž

$$\begin{aligned} a(x) &= x^8 + x^6 - 3x^4 - 3x^3 + 8x^2 + 2x - 5 \\ b(x) &= 3x^6 + 5x^4 - 4x^2 - 9x + 21. \end{aligned}$$

Zvolíme-li např. $\xi = 5$, pak dostaneme $a(5) = 404205, b(5) = 49876$. Pomocí Euklidova algoritmu v $\mathbb{Z}$ vypočteme $\gcd(404205, 49876) = 1$. Rádi bychom z tohoto výsledku dokázali fakt (jak se ukázalo např. v Příkladu 3.11, pravdivý), že polynomy $a(x), b(x)$ jsou nesoudělné.

Že nepůjde o nabízející se naivní vztah, ukazuje následující protipříklad. Vezmeme-li polynomy $(x-4)\cdot a(x)$ a $(x-4)\cdot b(x)$, snadno se vidí, že hodnoty těchto polynomů v bodě $\xi = 5$ jsou stejné jako hodnoty polynomů $a(x), b(x)$, tedy nesoudělné, přestože polynomy $(x-4)\cdot a(x)$ a $(x-4)\cdot b(x)$ mají na první pohled společného dělitele $x-4$.

Ukažme nyní na případě okruhu polynomů s celočíselnými koeficienty, jak budeme odvozovat tvar největšího společného dělitele polynomů na základě znalosti největšího společného dělitele hodnot těchto polynomů v určitém bodě. Předpokládejme, že máme polynomy $a(x), b(x) \in \mathbb{Z}[x]$ a číslo $\xi \in \mathbb{Z}$. Označme pro tuto chvíli $\gamma = \gcd(a(\xi), b(\xi))$ a $c(x) = \gcd(a(x), b(x))$. Protože zobrazení (tzv. *evaluace* neboli vyhodnocení), přiřazující polynomu jeho hodnotu v bodě ξ je homomorfismus, víme, že nutně platí $c(\xi) \mid \gamma$. Budeme na chvíli předpokládat, že dokonce $c(\xi) = \gamma$ a ukážeme, jak ze znalosti γ získat polynom $c(x)$ za předpokladu, že je ξ dostatečně velké (později bude specifikováno). Vyjádříme γ v jeho ξ-adickém zápise, tj. ve tvaru

$$\gamma = \gamma_0 + \gamma_1\xi + \gamma_2\xi^2 + \cdots + \gamma_d\xi^d,$$

kde pro jednotlivé členy rozvoje platí $-\xi/2 < \gamma_i \leq \xi/2$ (tzv. *symetrická reprezentace modulo* ξ) a d je nejmenší přirozené číslo, pro které platí $\xi^{d+1} > 2|\gamma|$. Ukážeme, že při splnění jistých podmínek bude

$$c(x) = \gamma_0 + \gamma_1 x + \gamma_2 x^2 + \cdots + \gamma_d x^d.$$

Ilustrujme opět tento postup na konkrétním příkladě.

Příklad 4.14. Použijme nyní polynomy z Příkladu 3.14 (je to z toho důvodu, že ve standardně užívaném příkladu jsou polynomy ireducibilní). Zde

$$\begin{aligned} a(x) &= 30x^3 - 5x^2 + 5x - 10 \\ b(x) &= -9x^2 - 39x + 30 \end{aligned}$$

jsou polynomy jedné proměnné s celočíselnými koeficienty. Viděli jsme již dříve, že je výhodné počítat zvlášť GCD primitivních částí polynomu, proto to tak proveďme i nyní (v [3, Př. 7.22] je ukázáno, proč je to v tomto případě dokonce nutností). Máme nyní

$$\begin{aligned} a_1(x) &= \mathrm{pp}(a(x)) = 6x^3 - x^2 + x - 2 \\ b_1(x) &= \mathrm{pp}(b(x)) = 3x^2 + 13x - 10, \end{aligned}$$

přitom $\operatorname{cont}(a(x)) = 5, \operatorname{cont}(b(x)) = -3$. Zvolíme-li v tomto případě $\xi = 30$, dostaneme

$$a_1(30) = 161128 \quad \text{a} \quad b_1(30) = 3080,$$

což dává $\gcd(161128, 3080) = 88$. Provedeme-li rozvoj čísla 88 v 30-adické soustavě (při použití symetrické reprezentace), dostaneme $88 = 3 \cdot 30 - 2$, což dává

$$\gcd(\operatorname{pp}(a(x)), \operatorname{pp}(b(x))) = 3x - 2.$$

Protože $\gcd(\operatorname{cont}(a(x)), \operatorname{cont}(b(x))) = 1$, vidíme, že je to i největší společný dělitel polynomů $a(x)$ a $b(x)$ (kontrolu poskytuje zmíněný Příklad 3.14).

Zobecnění z případu polynomů jedné proměnné na polynomy více proměnných je přímočaré — postupně se volí vyhodnocovací body v jednotlivých proměnných a v opačném směru se v obráceném pořadí interpoluje.

Jak jsme dříve poznamenali, předpoklad $\gamma = c(\xi)$ je pouze naším toužebným přáním. Ve skutečnosti nemusí obecně platit vztah $\gamma \mid c(\xi)$. Ukážeme ale, že pokud bude pro polynom $c'(x)$ generovaný interpolací z čísla γ platit $c'(x) \mid a(x)$ i $c'(x) \mid b(x)$, pak je $c'(x)$ největším společným dělitelem $a(x)$ a $b(x)$, tj. $c'(x) = c(x)$. Zároveň ukážeme slibovanou podmínku na velikost čísla ξ, která je vyjádřena pomocí tzv. *normy* polynomu více proměnných s celočíselnými koeficienty, kterou budeme pro naše účely rozumět největší celočíselný koeficient v absolutní hodnotě (normu polynomu $a(\boldsymbol{x})$ budeme značit $\|a(\boldsymbol{x})\|$).

Věta 4.15. *Buďte $a(\boldsymbol{x}), b(\boldsymbol{x}) \in \mathbb{Z}[x_1, x_2, \ldots, x_k]$ nenulové polynomy, které jsou primitivní* [3] *vzhledem k $\mathbb{Z}$. Buď dále $\xi \in \mathbb{Z}$ číslo, splňující*

$$\xi > 1 + 2\min(\|a(\boldsymbol{x})\|, \|b(\boldsymbol{x})\|).$$

Položme $\gamma(\boldsymbol{x}) = \gcd(a(x_1, \ldots, x_{k-1}, \xi), b(x_1, \ldots, x_{k-1}, \xi)) \in \mathbb{Z}[x_1, \ldots, x_{k-1}]$ a jako $c(\boldsymbol{x})$ označme polynom vytvořený pomocí ξ-adické reprezentace $\gamma(\boldsymbol{x})$. Označíme-li jako $\operatorname{pp}(c(\boldsymbol{x}))$ polynom vzniklý vydělením polynomu $c(\boldsymbol{x})$ největším společným dělitelem jeho celočíselných koeficientů, pak je $\operatorname{pp}(c(\boldsymbol{x}))$ největším společným dělitelem polynomů $a(\boldsymbol{x})$ a $b(\boldsymbol{x})$ právě tehdy, když

$$\operatorname{pp}(c(\boldsymbol{x})) \mid a(\boldsymbol{x}) \quad \textit{a} \quad \operatorname{pp}(c(\boldsymbol{x})) \mid b(\boldsymbol{x}).$$

K důkazu Věty 4.15 budeme potřebovat dvě pomocná lemmata. První z nich je známá podmínka na velikost kořenů polynomů, která bude potřeba v důkazu druhého lemmatu, klíčového pro vlastní důkaz věty.

[3] Tj. GCD jejich celočíselných koeficientů je roven 1.

Lemma 4.16. *Je-li $a(x) = a_n x^n + \cdots + a_0 \in \mathbb{C}[x]$, kde $a_n \neq 0, n \geq 1$, polynom, pak pro libovolný kořen $\alpha \in \mathbb{C}$ tohoto polynomu platí*

$$|\alpha| < 1 + \frac{A}{|a_n|},$$

kde $A = \max(|a_{n-1}|, \ldots, |a_1|, |a_0|)$.

Důkaz. Tvrzení dokážeme sporem. Předpokládejme naopak, že $|\alpha| \geq A/|a_n| + 1$. Pak

$$A \frac{|\alpha|^n}{|\alpha| - 1} \leq |a_n||\alpha|^n.$$

Protože je podle předpokladu $|\alpha| > 1$, máme

$$|a_n||\alpha|^n \geq A \frac{|\alpha|^n}{|\alpha| - 1} > A \frac{|\alpha|^n - 1}{|\alpha| - 1} = A(|\alpha|^{n-1} + |\alpha|^{n-2} + \cdots + 1) \geq$$
$$|a_{n-1}||\alpha|^{n-1} + |a_{n-2}||\alpha|^{n-2} + \cdots + |a_0| \geq |a_{n-1}\alpha^{n-1} + \cdots + a_{n-2}\alpha^{n-2} + \cdots + a_0|.$$

Celkem tedy dostáváme $|a_n \alpha^n + a_{n-1}\alpha^{n-1} + \cdots + a_0| > 0$, což je spor s předpokladem $a(\alpha) = 0$. □

Lemma 4.17. *Nechť $p(\boldsymbol{x}) \in \mathbb{Z}[x_1, \ldots, x_k]$ je nekonstantní polynom, $\delta \in \mathbb{N}$ a $j \in \{1, \ldots, k\}$ přirozené číslo. Splňuje-li $m \in \mathbb{Z}$ nerovnost*

$$|m| \geq \|p(x)\| + \delta + 1,$$

pak pro libovolný nekonstantní *polynom $q(\boldsymbol{x}) \in \mathbb{Z}[x_1, \ldots, x_k]$, splňující*

$$q(\boldsymbol{x}) \mid p(\boldsymbol{x}) \quad a \quad q(x_1, \ldots, x_{j-1}, m, x_{j+1}, \ldots, x_k) \in \mathbb{Z}$$

platí

$$|q(x_1, \ldots, x_{j-1}, m, x_{j+1}, \ldots, x_k)| > \delta.$$

Důkaz. Uvažme nejprve případ polynomů jedné proměnné, případ polynomů více proměnných pak na tento případ snadno převedeme. Nechť $\alpha_i \in \mathbb{C}$ pro $i = 1, \ldots, n$ jsou všechny komplexní kořeny polynomu $p(x)$ a $a = \mathrm{lcoeff}(p(x)) \neq 0$. Pak rozklad polynomu $p(x)$ na lineární činitele má tvar

$$p(x) = a \cdot \prod_{i=1}^{n} (x - \alpha_i).$$

Podle Lemmatu 4.16 dostáváme

$$|\alpha_i| < 1 + \frac{\|p(x)\|}{|a|}.$$

Pro libovolné $m \in \mathbb{Z}$ dále zřejmě platí

$$p(m) \geq |a| \cdot \prod_{i=1}^{n} \Big| |m| - |\alpha_i| \Big|,$$

splňuje-li pak navíc $m \in \mathbb{Z}$ podmínku z tvrzení lemmatu, dostáváme

$$\Big| |m| - |\alpha_i| \Big| > \left| (\|p(x)\| + \delta + 1) - \left(1 + \frac{\|p(x)\|}{|a|}\right) \right| \geq \delta$$

a celkem

$$|p(m)| > |a|\delta^n \geq \delta.$$

Protože je polynom $q(x)$ dělitelem $p(x)$, jsou všechny jeho kořeny mezi čísly α_i a platí tedy rovněž $|q(m)| > \delta$.

Uvažujme nyní případ $k > 1$, tj. polynomy více proměnných. Snadno je vidět, že lze zvolit hodnoty $m_i \in \mathbb{Z}$ pro $i \neq j$ tak, že polynom $p_1(x) = p(m_1, \ldots, m_{j-1}, x, m_{j+1}, \ldots, m_k)$ jedné proměnné má hodnoty koeficientů, které splňují

$$\|p_1(x)\| \leq \|p(\boldsymbol{x})\|.$$

Splňuje-li nyní polynom $q(\boldsymbol{x}) \in \mathbb{Z}[x_1, \ldots, x_k]$ podmínky zadání, pak platí $q_1(x) = q(m_1, \ldots, m_{j-1}, x, m_{j+1}, \ldots, m_k) \mid p_1(x)$ a z již dokázaného případu polynomů jedné proměnné plyne $|q_1(m)| > \delta$. Protože však podle podmínek lemmatu $q(x_1, \ldots, x_{j-1}, m, x_{j+1}, \ldots, x_k) \in \mathbb{Z}$, je nutně $|q_1(m)| = |q(x_1, \ldots, x_{j-1}, m, x_{j+1}, \ldots, x_k)| > \delta$. Tím je tvrzení dokázáno. $\square$

Nyní můžeme přistoupit k vlastnímu důkazu věty.

Důkaz Věty 4.15. Nutnost podmínky z tvrzení věty, totiž fakt, že největší společný dělitel polynomů $a(\boldsymbol{x})$ a $b(\boldsymbol{x})$ je jejich společným dělitelem, je zřejmá. Dokážeme, že tato podmínka je i postačující.

Označme $p(\boldsymbol{x}) = \mathrm{pp}(c(\boldsymbol{x})) = c(\boldsymbol{x})/d$, kde $d \in \mathbb{Z}$ a $g(\boldsymbol{x}) = \gcd(a(\boldsymbol{x}), b(\boldsymbol{x}))$. Pak podle předpokladů máme $p(\boldsymbol{x}) \mid g(\boldsymbol{x})$ a existuje $h(\boldsymbol{x}) \in \mathbb{Z}[x_1, \ldots, x_k]$ tak, že $g(x) = p(x) \cdot h(x)$ a také $g(x_1, \ldots, x_{k-1}, \xi) = p(x_1 \ldots, x_{k-1}, \xi) \cdot h(x_1 \ldots, x_{k-1}, \xi)$. Protože $\gamma(\boldsymbol{x}) = c(x_1, \ldots, x_{k-1}, \xi)$, máme

$$g(x_1, \ldots, x_{k-1}, \xi) = \gamma(x)/d \cdot h(x_1, \ldots, x_{k-1}, \xi).$$

Nyní ale rovněž $g(\boldsymbol{x}) \mid a(\boldsymbol{x})$ a $g(\boldsymbol{x}) \mid b(\boldsymbol{x})$, z čehož plyne $g(x_1, \ldots, x_{k-1}, \xi) \mid \gamma(\boldsymbol{x})$ a tedy $\gamma(\boldsymbol{x})/d \cdot h(x_1, \ldots, x_{k-1}, \xi) \mid \gamma(\boldsymbol{x})$. Úpravou tohoto vztahu dostaneme $h(x_1, \ldots, x_{k-1}, \xi) \mid d$, což implikuje $h(x_1, \ldots, x_{k-1}, \xi) \in \mathbb{Z}$ a

$$|h(x_1, \ldots, x_{k-1}, \xi)| \leq d \leq \xi/2.$$

Na druhé straně, podle podmínky v tvrzení věty je

$$\xi/2 \geq 1 + \min(\|a(\boldsymbol{x})\|, \|b(\boldsymbol{x})\|)$$

a protože polynom $h(\boldsymbol{x})$ dělí $a(\boldsymbol{x})$ i $b(\boldsymbol{x})$, dostáváme podle Lemmatu 4.17, že pokud je $h(\boldsymbol{x})$ nekonstantní, platí

$$|h(x_1, \dots, x_{k-1}, \xi)| > \xi/2,$$

což je spor s předchozím. Je tedy nutně $h(\boldsymbol{x})$ konstantní polynom a protože je $g(\boldsymbol{x})$ primitivní (jako GCD primitivních polynomů), musí být nutně $h(\boldsymbol{x}) = \pm 1$, což znamená, že $p(\boldsymbol{x})$ je asociovaný s $g(\boldsymbol{x})$ a je tedy největším společným dělitelem $a(\boldsymbol{x})$ a $b(\boldsymbol{x})$. $\square$

Nyní již máme připraveny veškeré podmínky k tomu, abychom mohli prezentovat algoritmus GCDHEU, počítající největšího společného dělitele heuristickou metodou.

Algoritmus 4.18. K daným polynomům $a(\boldsymbol{x}), b(\boldsymbol{x}) \in \mathbb{Z}[x_1, \dots, x_k]$ se tento algoritmus pokusí rekurzivně vypočítat jejich největší společný dělitel procedurou GCDHEU$(a(\boldsymbol{x}), b(\boldsymbol{x}))$.

1. [Konstanty?] Jsou-li $a(\boldsymbol{x}), b(\boldsymbol{x})$ konstanty, pak vrať jejich GCD vypočítaný např. pomocí Euklidova algoritmu. Jinak označ jako x poslední proměnnou a polož $\xi \leftarrow 2 \cdot \min(\|a(\boldsymbol{x})\|, \|b(\boldsymbol{x})\|) + 2, n \leftarrow 0$.

2. [Neúspěch?] Pokud je $n > 5$ vrať jako výsledek „fail“. Je-li $\operatorname{sizeof}(\xi) \cdot \max(\deg_x(a(\boldsymbol{x})), \deg_x(b(\boldsymbol{x}))) > 5000$, skonči celý výpočet neúspěchem.

3. [Rekurze] Buď $\gamma(\boldsymbol{x}) \leftarrow \mathrm{GCDHEU}(a(x_1, \dots, x_{k-1}, \xi), b(x_1, \dots, x_{k-1}, \xi))$. Pokud je $\gamma(\boldsymbol{x}) =$ „fail“, jdi na 6, jinak polož $c(\boldsymbol{x}) \leftarrow 0, i \leftarrow 0$.

4. [Test dělení] Je-li $\gamma(\boldsymbol{x}) = 0$, polož $c(\boldsymbol{x}) \leftarrow \mathrm{pp}(c(\boldsymbol{x}))$ a pokud je $c(\boldsymbol{x})$ dělitelem $a(\boldsymbol{x})$ i $b(\boldsymbol{x})$, vrať $c(\boldsymbol{x})$ jako jejich největšího společného dělitele. Pokud není dělitelem, jdi na 6.

5. [Generuj $c(\boldsymbol{x})$] Buď $d(\boldsymbol{x})$ symetrický reprezentant polynomu $\gamma(\boldsymbol{x})$ modulo ξ, dále polož $c(\boldsymbol{x}) \leftarrow c(\boldsymbol{x}) + d(\boldsymbol{x}) \cdot x^i, \gamma(\boldsymbol{x}) \leftarrow (\gamma(\boldsymbol{x}) - d(\boldsymbol{x}))/\xi, i \leftarrow i + 1$ a jdi na 4.

6. [Nové ξ] Polož $\xi \leftarrow \mathrm{quo}(73794 \cdot \xi, 27011), n \leftarrow n + 1$ a jdi na 2.

K pochopení vlastností tohoto algoritmu je nutné učinit několik poznámek. Je důležité, že známe spodní hranici, od které se můžeme „spolehnout“ na zvolené ξ, protože s rostoucí velikostí ξ budou růst i hodnoty koeficientů po vyhodnocení polynomu v bodě $x_k = \xi$. Z tvrzení věty je patrné, že rozhodující měrou bude na velikost výsledných celých čísel působit počet proměnných původních polynomů. Snadno se ukáže, že velikost celých čísel, pro něž počítáme jejich největšího společného dělitele v $\mathbb{Z}$ běžným Euklidovým algoritmem je $O(\xi^{d^k})$, kde k je počet proměnných, d je stupeň každé proměnné a $x_k = \xi$ je první vyhodnocení počátečních polynomů. Proto je nutné do algoritmu zabudovat podmínku, která při neúnosném zvýšení koeficientů výpočet největšího společného dělitele touto metodou vzdá. U praktických problémů se ukazuje, že nejvýhodnější je použití této metody na polynomy jedné proměnné, případně polynomy přibližně čtyř proměnných s nepříliš vysokými koeficienty.

V případě, že neuspěl test dělení podle předchozí věty, je možné volit jiný vyhodnocovací bod ξ. Protože jde o pravděpodobnostní algoritmus, je výhodné volit novou hodnotu ξ tak, aby byly hodnoty polynomů v těchto bodech co nejméně korelovány. Nejpřirozenější volbou je násobení čísla ξ nějakým číslem větším než jedna, které nebude celé (aby v rozkladu na prvočísla jednotlivých čísel ξ nevystupovaly společné faktory) a vzetím celé části ze součinu. Například v knize [3] autoři volí násobení čísla ξ aproximací zlatého řezu, tj. hodnotou 73794/27011, autor implementace pro Axiom (kde se ovšem omezují na případ polynomů jedné proměnné) zvolili násobení číslem 832040/317811, přičemž dbají navíc na to, aby výsledné číslo ξ nebylo sudé, protože jako počáteční je voleno číslo sudé.

4.4 Henselův (EZGCD) algoritmus

Jak jsme již naznačili, algoritmy modulárního (resp. pravděpodobnostního) typu bývají zpravidla úspěšnější než algoritmy založené na pseudodělení polynomů a zbytkových posloupnostech. V naprosté většině výpočtů s malým počtem proměnných dobře vyhoví heuristická metoda a v případě, že tato selže, je možné aplikovat modulární metodu. Jak je však z tvaru této metody vidět, v případě mnoha proměnných, jde-li navíc o *řídké* polynomy, provádí se zde příliš mnoho (jednoduchých) výpočtů v euklidovských okruzích. Jedním z možných překlenutí tohoto problému je naznačený algoritmus *řídkého modulárního GCD*. Jinou metodou (která je například v systému Maple použita, kdykoli selže heuristická metoda) je algoritmus, který bývá nazýván *Extended Zassenhaus GCD* nebo *Henselův algoritmus*. V této části nastíníme princip této metody a její výhody, podrobnou teorii nutnou pro odvození algoritmu nebudeme z rozsahových důvodů budovat a žádostivého čtenáře odkážeme na základní informace a literaturu k tomuto tématu uvedenou v učebnici [3].

Základním teoretickým výsledkem, aplikovaným v tomto algoritmu je tzv. *Henselovo lemma*, které zde uvedeme bez důkazu.

Věta 4.19 (Henselovo lemma). *Buďte $u(x), v_1(x), w_1(x)$ polynomy s celočíselnými koeficienty takové, že $v_1(x)$ a $w_1(x)$ jsou nesoudělné modulo prvočíslo p a*

$$u(x) = v_1(x)w_1(x) \pmod p$$

Pak pro libovolné $e \geq 1$ existují polynomy $v_e(x), w_e(x)$ s celočíselnými koeficienty, splňující $v_e(x) = v_1(x) \pmod p$, $w_e(x) = w_1(x) \pmod p$ a

$$u(x) = v_e(x)w_e(x) \pmod{p^e}.$$

Navíc jsou tyto polynomy určeny jednoznačně modulo p^e.

Jak plyne z uvedeného lemmatu, hlavní aplikace Henselova lemmatu je při faktorizaci polynomů. Vypočte-li se faktorizace modulo prvočíslo p , pak ji lze za použití Henselova lemmatu přenést na rozklad modulo p^e kde exponent e je dostatečně velký, aby shora omezil dvojnásobek možné velikosti jednotlivých koeficientů výsledného rozkladu. Přitom je tento „přenos" velmi efektivní, neboť polynomy $v_{e+1}(x)$ a $w_{e+1}(x)$ dostaneme z $v_e(x)$ a $w_e(x)$ jedním krokem tzv. *p-adické interpolace*.

Zobecněním předchozí věty je Henselovo lemma pro polynomy více proměnných, které navíc kromě výpočtů modulo používá vyhodnocování jednotlivých proměnných v celočíselných hodnotách (vlastní věta by byla naprosto analogická, protože lze oba postupy shrnout pod společný algebraický rámec hledání inverzního zobrazení k modulárnímu, resp. vyhodnocovacímu homomorfismu).

Základní idea, využívající Henselova lemmatu k výpočtu největšího společného dělitele polynomů více proměnných s celočíselnými koeficienty, je následující:

- polynomy $a(x)$ a $b(x)$ redukujeme pomocí výpočtů modulo a vyhodnocování v bodech na polynomy $a_1(x), b_1(x)$ jedné proměnné nad konečným tělesem a zde vypočteme GCD $c_1(x)$

- v obvyklém případě, kdy je takto vypočítaný $c_1(x)$ nesoudělný s jedním z polynomů $a_1(x)/c_1(x)$ nebo s $b_1(x)/c_1(x)$, můžeme použít Henselova lemmatu na posun této faktorizace modulo p na faktorizaci modulo p^e pro dostatečně velké e

- tuto faktorizaci polynomů jedné proměnné „posuneme“ na faktorizaci původních polynomů pomocí Henselova algoritmu pro polynomy více proměnných
- otestujeme, zda „posunutý“ největší společný dělitel $c_e(x)$ dělí polynomy $a(x)$ a $b(x)$ a v případě, že ano, je hledaným GCD

Tato metoda má následující dobré vlastnosti:

- případ, že polynomy $a(x)$ a $b(x)$ jsou nesoudělné, odhalí velmi brzy stejně jako ostatní modulární metody tak, že polynom $c_1(x)$ bude konstantní
- v případě, že volíme vyhodnocování v nule, kdykoli je to jen možné, je zachována řídká struktura původních polynomů

Mezi implementační detaily této metody patří podobně jako u předchozích metod problém, jak se vyrovnat s *nešťastnými homomorfismy*, jak omezit velikost koeficientů a problém s vedoucími koeficienty. Zásadním problémem pak je, jak se vypořádat s těmi řídkými případy, kdy je hledaný GCD c(x) soudělný s $a(x)/c(x)$ i $b(x)/c(x)$. Pak je nutno místo počítání GCD polynomů nad konečným tělesem pokročit dále ve faktorizaci obrazů našich polynomů. Míra této faktorizace se pak liší v návrzích jednotlivých autorů (původní autoři této metody, pánové Moses a Yun doporučují tzv. *square-free* rozklad polynomu $a(x)$ nebo $b(x)$ — podrobněji viz např. [3]).

Kapitola 5

Praktická implementace jednotlivých algoritmů

V této kapitole se zmíníme o praktické realizaci uvedených algoritmů. Zaměříme se přitom na jejich implementaci v systémech Maple V R3 a Axiom verze 2.0. Dále budeme diskutovat jejich použitelnost na různé typy polynomů.

5.1 Maple

Jak jsme již poznamenali v kapitole o historii systémů počítačové algebry, funkce v Maplu se dělí na tzv. *interní*, které tvoří jádro Maplu a jsou kvůli efektivitě naprogramovány v jazyce C (např. funkce `igcd` počítající největšího společného dělitele dvou celých čísel), a *externí*, které jsou naprogramovány ve vyšším programovacím jazyce Maplu. Tyto externí funkce jsou pak obvykle slinkovány do knihovny `maple.lib`. Dalším zdrojem algoritmů, je tzv. *share* knihovna, která obsahuje algoritmy vzniklé mimo vývojářský tým Maplu. Důležitým prvkem Maplu je to, že umožňuje získat zdrojový kód externích procedur (pomocí nastavení `verboseproc`). Navíc můžeme získat informace o průběhu výpočtu pomocí nastavení `infolevel` (algoritmy psané se zřetelem na tuto vlastnost v sobě obsahují příkazy, které při určité hodnotě `infolevel` vypíší některé důležité informace o průběhu algoritmu).

Na obrázku 5.1 je uveden fragment[1] programu, který vykonává Maple při výpočtu největšího společného dělitele polynomů. Z programu můžeme vyčíst, že při hledání největšího společného dělitele polynomů jedné proměnné se

[1]Důležité implementační a optimalizační detaily byly kvůli stručnosti vynechány. Dychtivý čtenář může kompletní kód procedury `gcd` získat v Maplu po zadání příkazů `"interface(verboseproc=2):print(gcd);"`

```
gcd:=proc(aa,bb,cofa,cofb)
local a,b,inda,indb,inds,ta,tb,res,x,y,GCD,II,Z;
options `Copyright 1992 by the University of Waterloo`;
    do
        ta := icontent(a); divide(a,ta,'a');
        tb := icontent(b); divide(b,tb,'b');
        inds := indets(a) union indets(b);
        inda := indets(a); indb := indets(b);
        if divide(b,a) then GCD := a; break
        elif degree(a,inda) < 2 then GCD := 1; break
        elif divide(a,b) then GCD := b; break
        elif degree(b,indb) < 2 then GCD := 1; break
        fi;
        inds := inda intersect indb;
        if inds = {} then GCD := 1; break fi;
        if nops(inda union indb) = 1 then
            x := op(1,inda union indb); GCD := `gcd/gcdchrem1`(a,b,x); break
        fi;

        userinfo(2,gcd,`trying the heuristic routine gcdheu`);
        GCD := `gcd/gcdheu1`(a,b);
        if GCD <> FAIL then break fi;
        if nops(inda union indb) = 2 then
            x := op(1,inda union indb); y := op(2,inda union indb);
            GCD := `gcd/gcdchrem2`(a,b,x,y);
            break
        fi;

        userinfo(2,gcd,`entering extended hensel`);
        GCD := `gcd/gcdeh`(a,b);
        userinfo(2,gcd,`exiting extended hensel`)
        break
    od;
    GCD
end
```

Obrázek 5.1: Schéma výpočtu GCD v Maplu

```
gcdheu:=proc(aa,bb)
local a,b,vars,x,ksi,gamma,G,i,ca,cb;

        userinfo(3,gcdheu,`entered with`,aa,bb);
        a:=expand(aa);b:=expand(bb);
        `mod`:=mods;

        vars:=indets(a) union indets(b);
        if nops(vars)=0 then RETURN(igcd(a,b)) else x:=vars[1] fi;
        ca:=icontent(a);a:=a/ca;
        cb:=icontent(b);b:=b/cb;
        norma:=max(op(map(abs,[coeffs(a)])));
        normb:=max(op(map(abs,[coeffs(b)])));
        ksi:=2*min(normb,normb)+2;
        userinfo(5,gcdheu,`ksi=`,ksi);

        to 6 do
        if (length(ksi) * max(degree(a,x),degree(b,x))) > 5000 then
           ERROR(`coefficients too big`);       # return_to_top_level(fail);
        fi;

        gamma:=gcdheu(subs(x=ksi,a),subs(x=ksi,b));
        userinfo(5,gcdheu,`gamma=`,gamma);

        if gamma<> fail_flag then
                        # generate polynom G from ksi-adic form of gamma
                G:=0;
                for i from 0 while gamma<> 0 do
                        g.i:= gamma mod ksi;
                        G:=G+g.i*x^i;
                        gamma:=(gamma-g.i)/ksi
                od;

                G:=G/icontent(G);
                userinfo(5,gcdheu,`candidate for gcd`,G);
                if divide(a,G) and divide(b,G) then
                        userinfo(5,gcdheu,`... satisfied`);
                        RETURN(igcd(ca,cb)*G)
                else
                        userinfo(5,gcdheu,`... failed`);
                fi;
        fi;
        ksi:=iquo(ksi*73794,27011);
        userinfo(5,gcdheu,`ksi changed to `,ksi);
        od;

        RETURN(fail_flag);
end:
```

Obrázek 5.2: Heuristická metoda na výpočet GCD provede procedura `gcdchrem1`, která je implementací modulárního algoritmu pro případ jedné proměnné (chrem = ***ch****inese* ***rem****ainder*). V případě polynomů více proměnných se pak nejprve aplikuje heuristický algoritmus, který je velmi efektivní pro polynomy s malým počtem proměnných (a rozumnou velikostí koeficientů). Algoritmus `gcdheu` v sobě obsahuje, podobně jako náš algoritmus 4.18 (viz také obr.5.2), test na velikost celých čísel, používaných ve výpočtu celočíselného GCD. V případě neúspěchu pak skončí nezdarem a hlavní rutina počítá největšího společného dělitele pomocí algoritmu EZGCD (Henselův algoritmus) nebo (v případě polynomů dvou proměnných) speciální variantou modulárního algoritmu. Vidíme tedy, že z praktického hlediska je zřejmě výhodnější použití algoritmů založených na modulárních technikách než výpočet GCD pomocí polynomiálních zbytkových posloupností.

Maple obsahuje rovněž implementaci algoritmu 4.3, která je pod názvem `subres` přítomna v knihovně *share*. Část tohoto balíku, která byla v této práci diskutována, je možné vidět na obrázku 5.3 (druhou část tvoří výpočet GCD v tzv. algebraickém číselném tělese).

5.2 Axiom

V systému počítačové algebry Axiom je situace poněkud odlišná. Z povahy systému vyplývá, že Axiom netestuje typ polynomů (resp. počet proměnných), nad kterými chce uživatel počítat největšího společného dělitele, ale pouze kontroluje, zda typ volané funkce, implementující algoritmus největšího společného dělitele, odpovídá argumentům. Standardním algoritmem, který se v Axiomu použije, je-li zavolána funkce `gcd`, je implementace Henselova algoritmu v balíku `PolynomialGcdPackage`. Tato funkce pak umožňuje použití tohoto algoritmu na libovolný okruh polynomů nad euklidovským okruhem. V Axiomu je rovněž (v balíku `HeuGcd`) implementován algoritmus na výpočet GCD heuristickou metodou pro polynomy jedné proměnné nad okruhem $\mathbb{Z}$ celých čísel. Tento algoritmus se standardně volá v případě, že uživatel žádá výpočet největšího společného dělitele seznamu polynomů (dá se tedy např. vynutit výpočet GCD polynomů $f(x), g(x) \in \mathbb{Z}[x]$ heuristickou metodou, použijeme-li volání `gcd([f,g])`). Další alternativou je funkce `modularGcd` z balíku `InnerModularGcd`, která aplikuje modulární algoritmus na polynomy jedné proměnné nad euklidovským okruhem.

Vidíme tedy, že algoritmy, které jsou implementovány v obou systémech, jsou v podstatě tytéž, rozdíl je pouze ve filozofii systémů v tom, jakou míru kontroly má uživatel nad voláním konkrétních algoritmů. Jistou výhodou systému Maple je to, že standardní procedura `gcd` „rozhodne“ o použitém

```
#--> subres(a,b,x)
#  G. Labahn and M. Monagan July 1991.

`subres/rational` := proc(a,b,x)
local c,d,j,u,v,r,g,h,du,ddu,dv,ddv,S,s,t;

        u := expand(a);
        v := expand(b);

        if degree(u,x) < degree(v,x) then
           t := u; u := v; v := t; s := (-1)^(degree(u,x)*degree(v,x));
        else
           s := 1;
        fi;

        du := degree(u,x);
        dv := degree(v,x);

if printlevel > 2 then
lprint(`subres: degrees of input polynomials`,du,dv) fi;

        S := collect(u,x);
        ddu := du; ddv := dv;
        if u = 0 or v = 0 then RETURN(S) fi;
        if du = 0 and dv = 0 then RETURN(1) fi;
        if dv = 0 then RETURN(u,v,expand(v^du)) fi;

        c := 1;  g := 1;  h := 1;
        while dv > 0 do
                d := du-dv;
                c := c*(-1)^(du*dv);
                r := prem(u,v,x);
                u := v; v := r; du := dv; dv := degree(r,x);
                j := degree(u,x);
                S := S, collect(s^((ddu-j)*(ddv-j))*u,x);

if printlevel > 2 then
lprint(`subres: degree of pseudo remainder`,degree(u,x)) fi;

                divide(v,g*h^d,'v');
                g := coeff(u,x,du);
                if d = 1 then h := g else divide(g^d,h^(d-1),'h') fi;
        od;
        if du = 1 then RETURN( S, expand(s^(ddu*ddv)*c*v) ) fi;
        divide(v,h,'r');
        S, expand(s^(ddu*ddv)*c*v*r^(du-1));
end:
```

Obrázek 5.3: Implementace algoritmu na výpočet GCD pomocí subresultantů algoritmu, na druhé straně Axiom umožňuje použití těchto algoritmů na mnohem větší třídě okruhů polynomů.

5.3 Použitelnost jednotlivých algoritmů

V rámci testů implementace jednotlivých algoritmů v systému Maple byly provedeny také časové testy. Porovnávány byly implementace jednotlivých algoritmů v Maplu a to tak, že jako argumenty těmto procedurám byly předkládány „takřka" náhodné polynomy. Libovolné náhodné polynomy nemá příliš smysl testovat z důvodů, které jsme naznačili v části 4.2, totiž, že dva náhodně vybrané polynomy jsou s největší pravděpodobností nesoudělné a tento fakt algoritmy založené na modulárním principu velmi rychle odhalí. Proto jsme pro testování uvažovali polynomy $p(x) \cdot q(x)$ a $q(x) \cdot r(x)$, kde $p(x), q(x)$ a $r(x)$ byly polynomy, generované funkcí Maplu `randpoly` (a tedy takřka vždy po dvou nesoudělné). Tato funkce poskytuje široké možnosti pro testování, dovoluje totiž zvolit interval, ve kterém se nacházejí hodnoty koeficientů, počet proměnných, stupeň, to, je-li polynom řídký či hustý apod.

Výsledky testů potvrdily teoretické předpoklady o jednotlivých algoritmech i některé domněnky, uvedené u jednotlivých algoritmů, zejména totiž ten fakt, že algoritmy založené na modulárních metodách jsou obecně podstatně efektivnější než algoritmy založené na polynomiálních zbytkových posloupnostech (kromě případu, kdy je tato posloupnost velmi krátká). Např. při testu polynomů dvou proměnných stupně 100, které mají koeficienty o sto cifrách, dospěl modulární algoritmus (implementace `gcdchrem2`) k výsledku řádově po několika minutách, zatímco výpočet pomocí subresultantu trval řádově (alespoň[2]) hodiny. Možnost použití algoritmu na výpočet GCD pomocí subresultantů se tak rýsuje zřejmě pouze u polynomů speciálního typu, kde se dá očekávat nízký počet pseudodělení (tj. například tehdy, víme-li předem, že dané polynomy mají velkého společného dělitele).

Ukázalo se rovněž, že v případech, kdy polynomy splnily podmínku na velikost koeficientů a stupeň proměnných, kladenou v heuristickém algoritmu, je tento algoritmus jednoznačně nejvýhodnější (pouze v případě polynomů jedné proměnné je často výhodnější použít modulární algoritmus, který pak obecně vede na výpočet celočíselného GCD s menšími argumenty).

[2] Při tomto testu neměl autor dostatek trpělivosti.

Literatura

[1] Zenon Ivanovich Borevich a Igor Rostislavovich Shafarevich. *Number Theory.* Academic Press, New York, NY, USA, 1966.

[2] Henri Cohen. *A Course in Computational Algebraic Number Theory.* Number 138 in Graduate texts in Mathematics. Springer-Verlag, Berlin, Germany, 1993.

[3] Keith O. Geddes, Stephen R. Czapor a George Labahn. *Algorithms for Computer Algebra.* Kluwer Academic Publishers, Boston, USA, třetí vydání, 1994.

[4] André Heck. *Introduction to Maple.* Springer-Verlag, Berlin, Germany, 1993.

[5] Keneth Ireland a Michael Rosen. *A Classical Introduction to the Modern Number Theory.* Number 84 in Graduate Texts in Mathematics. Springer-Verlag, 1990.

[6] Richard D. Jenks a Robert S. Sutor. *AXIOM: the scientific computation system.* Springer-Verlag, Berlin, Germany, 1992.

[7] Donald E. Knuth. *Seminumerical Algorithms*, volume 2 of *The Art of Computer Programming.* Addison-Wesley, Reading, MA, USA, druhé vydání, 1981.

[8] Saunders Mac Lane a Garret Birkhoff. *Algebra.* Alfa, Bratislava, SR, druhé vydání, 1974.