

Übungen zur Vorlesung Computeralgebra
Blatt 1

Prof. Dr. Klaus Madlener

1. Aufgabe: [Ringe und Körper]

Wir wollen einige einfache Sätze zeigen:

- a) Sei R ein Integritätsbereich, ferner seien $a, b \in R^*$ (die Menge der von 0 verschiedenen Elemente von R). Es sind a und b genau dann assoziiert, wenn es ein $u \in E(R)$ (die Einheitengruppe von R) mit $a = ub$ gibt.
- b) Sei R ein euklidischer Ring und sei I ein Ideal von R . Ist $0 \neq a \in I$, so ist genau dann $I = aR$, wenn $v(a) \leq v(b)$ für alle $b \in I \setminus \{0\}$ ist (wobei v die Bewertungsfunktion von R sei).
Zeigen Sie, dass weiter folgt, dass R ein Hauptidealring ist (d. h., dass alle Ideale von R Hauptideale sind).
- c) Ist K ein Körper, so ist $K[x]$ ein euklidischer Ring.
- d) Ist K ein Körper und L eine Erweiterung von K (also ein Körper, der K als Teilkörper enthält), ist weiter $f \in K[x]$ und $\ell \in L$ eine Nullstelle von f , so ist f in $L[x]$ durch $x - \ell$ teilbar.

2. Aufgabe: [Polynomdivision]

Wir untersuchen die Laufzeit der klassischen Polynomdivision mit Rest. Gegeben sei folgender Algorithmus:

function POLYQUOREM(a, b)

$a = \sum_{0 \leq i \leq n} a_i x^i$, $b = \sum_{0 \leq i \leq m} b_i x^i \in R[x]$, R ist ein kommutativer Ring

mit 1, alle $a_i, b_i \in R$, b_m ist eine Einheit in R und $n \geq m \geq 0$.

Ausgabe: $q, r \in R[x]$ mit $a = qb + r$ und $\deg r < m$ oder $r = 0$.

$r \leftarrow a$

for $i \leftarrow n - m, n - m - 1, \dots, 0$ **do**

if $\deg(r) = m + i$ **then**

$q_i \leftarrow \text{lc}(r)/b_m$; $r \leftarrow r - q_i x^i b$

else $q_i \leftarrow 0$

end if

end for

return $q = \sum_{0 \leq i \leq n-m} q_i x^i$ und r

end function

Nehmen Sie an, dass ein Polynom $p = \sum_{0 \leq i \leq k} p_i x^i$ vom Grad k durch eine dichte Darstellung gegeben sei, d. h. im Wesentlichen durch einen Koeffizientenvektor $\vec{p} = (p_0, \dots, p_k)$.

Geben Sie die Laufzeit, gemessen in der Anzahl von Ringoperationen in R , im schlechtesten Fall in Abhängigkeit von n und m an. Sind q und r im Allgemeinen eindeutig? Beweis!

3. Aufgabe: [Diophantische Gleichungen]

Gibt es $s, t \in \mathbb{Z}$, so dass $24s + 14t = 1$ bzw. so dass $61s + 37t = 56$? Geben Sie jeweils alle möglichen Lösungen an.

Zeigen Sie allgemeiner: Die lineare diophantische Gleichung $ax + by = c$ mit $a, b, c \in \mathbb{Z}$ ist genau dann (in \mathbb{Z}) lösbar, wenn für $d = \text{GGT}(a, b)$ gilt: $d|c$. Ist in diesem Fall (x_0, y_0) eine spezielle Lösung, dann ist

$$\{(x_0 + k \cdot \frac{b}{d}, y_0 - k \cdot \frac{a}{d} \mid k \in \mathbb{Z}\}$$

die Menge alle Lösungen. (Was bedeutet eigentlich die Schreibweise $\frac{a}{d}$ in diesem Zusammenhang?)

4. Aufgabe: [GGT]

a) Berechnen Sie größte gemeinsame Teiler von $f = x^5 + x^4 + x^3 - x^2 - x + 1$ und $g = x^3 + x^2 + x + 1$ ($f, g \in \mathbb{Z}_p[x]$) für $p = 3$ sowie $p = 5$. Berechnen Sie jeweils auch Polynome s und t mit $\text{ggT}(f, g) = sf + tg$.

b) Wir betrachten den folgenden ggT-Algorithmus nach J. Stein (war möglicherweise aber schon im antiken China bekannt):

```

1: function BINARYGCD( $u, v \in \mathbb{N}^+$ )                                ▷ Returns the g.c.d of  $u$  and  $v$ 
2:    $g \leftarrow 1$ 
3:   while ( $u \bmod 2 = 0$ )  $\wedge$  ( $v \bmod 2 = 0$ ) do
4:      $u \leftarrow u/2; v \leftarrow v/2; g \leftarrow 2g$ 
5:   end while
6:   while ( $u \neq 0$ ) do
7:     if ( $u \bmod 2 = 0$ ) then
8:        $u \leftarrow u/2$ 
9:     else if ( $v \bmod 2 = 0$ ) then
10:       $v \leftarrow v/2$ 
11:    else
12:       $t \leftarrow |u - v|/2$ 
13:      if  $u \geq v$  then
14:         $u \leftarrow t$ 
15:      else
16:         $v \leftarrow t$ 
17:      end if
18:    end if
19:  end while
20:  return  $g \cdot v$ 

```

21: **end function**

Zeigen Sie, dass dieser Algorithmus tatsächlich für Eingaben $u, v \in \mathbb{N}^+$ den Wert $\text{ggT}(u, v)$ berechnet, und zwar mit $O((\lambda(uv))^2)$ Bitoperationen im schlechtesten Fall. Dafür nehmen wir an, dass Zahlen aus \mathbb{N}^+ in Binärdarstellung gegeben sind und $\lambda(x)$ die Länge der Binärdarstellung (ohne führende Nullen) von x bezeichnet. Zeigen Sie abschließend, dass im schlechtesten Fall $O(\lambda(u)\lambda(v))$ Bitoperationen *nicht* ausreichen.

zu **Aufgabe1:**

a) „ \Rightarrow “: Sei $a \sim b$. Per Definition gilt $a|b$ und $b|a$, also $a = cb$ und $b = da$ für geeignete c, d . Dann gilt $a = cda$ und $b = dcb$, also sind c, d Einheiten.

„ \Leftarrow “: $b|a$ folgt sofort aus $a = ub$. Es gilt $b = u^{-1}a$, weil u Einheit ist, also auch $a|b$. Zusammen gilt $a \sim b$

b) „ \Rightarrow “: Sei $I = aR$. Für alle $b \in I$ existiert c mit $b = ca$. Also gilt $v(b) \geq v(a)$.

„ \Leftarrow “: Jedes $b \in I \setminus \{0\}$ lässt sich zerlegen in $b = aq + r$ mit $v(r) < v(a)$. Da $b \in I$ und $aq \in I$ ist auch $r \in I$. Dann muss aber $r = 0$ sein, sonst Widerspruch zur Voraussetzung. Folglich ist $b \in aR$.

Es bleibt zu zeigen, dass beliebige Ideale von R Hauptideale sind. Für $I = 0$ gilt die Behauptung. Für jedes andere Ideal sei $M = \{v(i) | i \in I \setminus \{0\}\}$. Die Menge M enthält ein minimales Element $m > 0$ und es gibt ein $a \in I$ mit $v(a) = m$. Dann gilt mit dem obigen, dass $I = aR$. Also sind alle Ideale in R Hauptideale.

c) Ist K ein Körper, so ist $K[x]$ ein euklidischer Ring. $K[x]$ ist Integritätsbereich (aus Definition der Addition und Multiplikation ersichtlich). Wir setzen $v(f) = \text{Grad}(f)$. Zu zeigen ist, dass für $f, g \in K[x]$ $q, r \in K[x]$ existieren mit $f = qg + r$, wobei $v(r) < v(g)$. Beweis durch Induktion über die Differenz der Grade von f und g . Falls $v(f) < v(g)$ gilt die Behauptung mit $q = 0$ und $r = f$. Sei also $v(f) \geq v(g)$. Sei $a_n x^n$ das Leitmonom von f und $b_m x^m$ das von g . Setze $u = f - a_n b_m^{-1} x^{n-m} g$. Dann ist $u = 0$ oder $v(u) < v(f)$. Nach Induktionsannahme gibt es $v, r \in K[x]$ mit $u = vg + r$ mit $r = 0$ oder $v(r) < v(g)$. Setze $q = a_n b_m^{-1} x^{n-m} + v$.

d) Ist K ein Körper und L eine Erweiterung von K (also ein Körper, der K als Teilkörper enthält), ist weiter $f \in K[x]$ und $\ell \in L$ eine Nullstelle von f , so ist f in $L[x]$ durch $x - \ell$ teilbar.

$L[x]$ ist euklidisch mit $v = \text{Grad}$. Also gilt für geeignete $q, r \in L[x]$ $f = q(x - \ell) + r$ mit $v(r) < v(x - \ell)$ oder $r = 0$. Dann ist $v(r) = 0$, also $r \in L$. Dann gilt $0 = f(\ell) = (\ell - \ell)q(\ell) + r = r$

zu **Aufgabe2:**

In jedem Schleifendurchlauf werden eine Division und m Multiplikationen und Additionen von Koeffizienten gemacht. Der Faktor x^i dient lediglich der Verschiebung, d.h. Berechnung der Indizes der Koeffizienten. Der höchste Koeffizient von r wird 0 und braucht deshalb nicht berechnet zu werden. Insgesamt ergeben sich $(2m + 1)(n - m + 1) \in O(n^2)$ Koeffizientenoperationen.

Seien $q, r, q', r' \in R[x]$ mit $a = qb + r$ und $a = q'b + r'$. Dann gilt $(q - q')b = (r - r')$. Diese Gleichung hat rechts ein Polynom mit Grad echt kleiner als dem von b . Das kann aber nur dann sein, wenn $q = q'$ ist, was sofort $r = r'$ zur Folge hat.

zu **Aufgabe3:**

$24s + 14t = 1$ ist nicht lösbar. Die linke Seite ist eine gerade Zahl, die rechte nicht. Für $61s + 37t = 56$ bestimmen wir mit dem EEA s und t , so dass $61s + 37t = \text{ggT}(61, 37) = 1$

	q	c	c_1	c_2	d	d_1	d_2
0		61	1	0	37	0	1
1	1	37	0	1	24	1	-1
2	1	24	1	-1	13	-1	2
3	1	13	-1	2	11	2	-3
4	1	11	2	-3	2	-3	5
5	5	2	-3	1	17	-28	
6	2	1	17	-28	0		

Also ist $s = 17 \cdot 56$ und $t = -28 \cdot 56$ eine Lösung. Um alle weiteren Lösungen zu bestimmen betrachtet man die homogene Gleichung $61s + 37t = 0$. Deren Lösungsmenge ist $\{(s, t) | s = 37r, t = -61r, r \in \mathbb{Z}\}$. Also hat man für die Ausgangsgleichung die Lösungsmenge $\{(s, t) | s = 37r + 17 \cdot 56, t = -61r - 28 \cdot 56, r \in \mathbb{Z}\}$.

Allgemein: Sei $d = \text{ggT}(a, b)$.

„ \Rightarrow “: Falls eine Lösung (x_0, y_0) existiert, dann gilt $d | ax_0 + by_0$, also $d | c$.

„ \Leftarrow “: Falls $d | c$ und $d = au + bv$ die vom EEA berechnete Zerlegung ist, dann gilt $c = a \cdot uc/d + b \cdot vc/d$, d.h. es gibt eine Lösung.

Die homogene Gleichung $ax + by = 0$ hat dieselbe Lösungsmenge wie $(a/d)x + (b/d)y = 0$. Da $\text{ggT}(a/d, b/d) = 1$ muss für Lösungen der homogenen Gleichung gelten, dass $(b/d) | x$ und $(a/d) | y$. Für Lösungen der homogenen Gleichung folgt als weitere Bedingung, dass $y = -(a/b)x$. Zusammen ergibt sich für Lösungen der allgemeinen Gleichung die gesuchte Lösungsmenge. Die Schreibweise a/d meint das Element e mit $a = ed$, das existiert, weil d per Definition ein Teiler von a ist.

zu **Aufgabe4:**

Der Algorithmus basiert auf folgenden Fakten:

- $\text{ggT}(u, v) = 2 \text{ggT}(u/2, v/2)$, falls u und v gerade.
- $\text{ggT}(u, v) = \text{ggT}(u/2, v)$, falls u gerade.
- $\text{ggT}(u, v) = \text{ggT}(|u - v|/2, v)$, falls u und v ungerade.

Zur Termination des Algorithmus: Die wesentliche Idee ist, zu zeigen, dass bei jedem Schleifendurchlauf sich das Produkt uv wenigstens halbiert. Der interessante Fall ist hier ab Zeile 12, wo das Maximum von u und v durch $|u - v|/2$ ersetzt wird. $|u - v|$ ist kleiner als das Maximum von u und v , die Hälfte ist also kleiner als dessen Maximum. Also gilt nach höchstens $1 + \log_2 uv$ Schritten, dass $uv = 0$. Höchstens u kann auf 0 gesetzt werden, weil $t = 0$ nur für den Fall $u = v$ eintreten, wenn beide ungerade sind. Also terminiert der Algorithmus. Die Korrektheit des Algorithmus folgt aus den o.g. Fakten.

In jedem Schleifendurchlauf sind eine konstante Anzahl von Additionen und Subtraktionen und Shifts von Zahlen der Länge $\lambda(u)$ und $\lambda(v)$ zu machen, die jeweils in linearer Zeit zu erledigen sind. Dabei sei $\lambda(x) = \lceil \log_2 x \rceil + 1$ die Anzahl der Binärstellen der Zahl x . (Alternative Def. siehe vzG: Computer Algebra 2.1) Zusammen sind dies $\lambda(u) + \lambda(v) + c$ Operationen für eine Konstante oder anders ausgedrückt: $\lambda(uv) + c$ Operationen. Für den gesamten Algorithmus ergeben sich $(1 + \log_2 uv)(\lambda(uv) + c) \in O((\lambda(uv))^2)$.

Wie oft kann bei einer Berechnung der Fall in Zeile 12 eintreten?

Sei $R(u, v)$ die Anzahl der Berechnungen dieses Falls, dann gilt $R(u, v) \leq \lfloor \log_2(u+v) \rfloor$.

Beweis: Induktion über $u + v$. Anfang: $u + v = 1$ gilt. Falls u und v gerade $R(u, v) \leq R(u/2, v/2) \leq \lfloor \log_2(u/2+v/2) \rfloor \leq \lfloor \log_2(u+v) \rfloor$. Falls u gerade und v ungerade $R(u, v) \leq R(u/2, v) \leq \lfloor \log_2(u/2 + v) \rfloor \leq \lfloor \log_2(u + v) \rfloor$. Analog u ungerade und v gerade. Falls u und v ungerade $R(u, v) = 1 + R((u-v)/2, v) \leq 1 + \lfloor \log_2((u-v)/2 + v) \rfloor = 1 + \lfloor \log_2((u+v)/2) \rfloor = \lfloor \log_2(u + v) \rfloor$

5. Aufgabe: [GGT]

- a) Zu Satz 2.13: Sei F_i für $i \in \mathbb{N}$ die i -te Fibonacci-Zahl (also $F_0 = 0$, $F_1 = 1$ und $F_i = F_{i-1} + F_{i-2}$ für $i \geq 2$). Zeigen Sie, dass $\lfloor F_{i+2}/F_{i+1} \rfloor = 1$ und $F_{i-1} = F_{i+1} \bmod F_i$ für alle $i \geq 2$.
- b) Sei $F[x]$ der euklidische univariate Polynomring über einem Körper F . Seien weiter $a, b \in F[x] \setminus \{0\}$ und $g = \text{ggT}(a, b) \in F[x]$. Zeigen Sie, dass es dann für jedes Polynom $c \in F[x]$ mit $g \mid c$ eindeutige Polynome $\sigma, \tau \in F[x]$ gibt, so dass $\sigma a + \tau b = c$ und $\deg(\sigma) < \deg(b) - \deg(g)$ gilt; wenn zudem noch $\deg(c) < \deg(a) + \deg(b) - \deg(g)$ ist, so gilt $\deg(\tau) < \deg(a) - \deg(g)$.
- c) Seien $a, b \in \mathbb{N}^+$ und $a > b$. Wir wollen entscheiden, ob es $i, j \in \mathbb{N}^+$ gibt, so dass $a^i = b^j$ ist. Betrachten Sie dazu folgendes Entscheidungsverfahren für dieses Problem:

Teste zuerst, ob $b \mid a$. Wenn nicht, so antworte „nein“. Ansonsten ersetze (a, b) durch $(a/b, b)$, wenn $a \geq b^2$, bzw. durch $(b, a/b)$, wenn $a < b^2$. Wenn durch Iterieren schließlich ein Paar $(a', 1)$ erreicht wird, antworte „ja“.

Zeigen Sie, dass dieses Verfahren das Problem für jede Eingabe korrekt löst (und terminiert) und im schlechtesten Fall $O(\lambda(a)^2)$ Bitoperationen benötigt.

6. Aufgabe: [Division in \mathbb{Z}]

- a) Wir betrachten noch einmal den Algorithmus zur Division mit Rest nicht negativer ganzer Zahlen zur Basis $b \geq 2$. Es seien $u = (u_0 \cdots u_n)_b$ sowie $v = (v_1 \cdots v_n)_b$ mit $\lfloor u/v \rfloor < b$. Es sei wie in der Vorlesung $\hat{q} = \min \left(\left\lfloor \frac{u_0 b + u_1}{v_1} \right\rfloor, b - 1 \right)$ die Schätzung für $q = \lfloor u/v \rfloor$ mit $u = qv + r$ und $0 \leq r < v$.

Zeigen Sie, dass $\hat{q} \geq q$ und für $v_1 \geq \lfloor b/2 \rfloor$ auch $\hat{q} - 2 \leq q$.

- b) Finden Sie ein Beispiel für u und v bei Basis 10, so dass die Notwendigkeit der bedingten Anweisung

if $(u_j \cdots u_{j+n})_b < \hat{q} \cdot (v_1 \cdots v_n)_b$ **then** $\hat{q} := \hat{q} - 1$

im Algorithmus aus der Vorlesung klar wird.

zu **Aufgabe5:**

a) Die Folge (F_i) ist monoton wachsend. Es gilt $F_{i+1} = F_i + F_{i-1}$. Es folgt für $i > 1$ $2F_i > F_{i+1} > F_i$. Also gilt $\lfloor F_{i+1}/F_i \rfloor = 1$. Für $i = 1$ gilt $F_{i+1} = F_i$.

Für die Zerlegung $F_{i+1} = qF_i + r$ bedeutet das, dass $q = 1$ und $r = F_{i-1}$ sein muss, also $F_{i-1} \cong F_{i+1} \pmod{F_i}$

b) Anders als im Fall der ganzen Zahlen, ist es bei $F[x]$ leichter als in \mathbb{Z} , die Bewertung der Bézout-Koeffizienten zu verkleinern. Der EEA liefert s und t mit $sa + tb = g$. Wenn nun $g|c$, erhält man $s\frac{c}{g}a + t\frac{c}{g}b = g\frac{c}{g} = c$. Sei s', t' die so erhaltene Lösung, d.h. $s'a + t'b = c$. Wir dividieren s' mit Rest durch b/g und erhalten q und σ mit $s' = q(b/g) + \sigma$, mit $\deg(\sigma) < \deg(b) - \deg(g)$. Setze $\tau = t + q(a/g)$ dann ist (σ, τ) eine Lösung der diophant. Gleichung, denn $\sigma a + \tau b = \sigma a + (t + q(a/g))b = (q(b/g) + \sigma)a + tb = sa + tb = c$.

Um eine obere Schranke des Grads von τ zu erkennen, schreibt man $\tau = (c - \sigma a)/b$. Das heißt, dass $\deg(\tau) = \deg(c - \sigma a) - \deg(b)$. Wenn $\deg(c) \geq \deg(\sigma a)$, dann $\deg(\tau) \leq \deg(c) - \deg(b) < \deg(a) - \deg(g)$ unter der zusätzlichen Voraussetzung. Wenn $\deg(c) < \deg(\sigma a)$, dann gilt $\deg(\tau) = \deg(\sigma a) - \deg(b) = \deg(a) + \deg(\sigma) - \deg(b) < \deg(a) - \deg(g)$ mit der gezeigten Schranke für $\deg(\sigma)$.

Angenommen es gäbe zwei verschiedene Lösungen (σ_1, τ_1) und (σ_2, τ_2) . Durch Subtraktion der durch g dividierten Ausgangsgleichung erhält man $(\sigma_1 - \sigma_2)(a/g) = -(\tau_1 - \tau_2)(b/g)$. Da a/g und b/g teilerfremd sind, muss $b/g | (\sigma_1 - \sigma_2)$ gelten. Es gilt aber auch die gezeigte Gradbedingung, so dass $\deg(\sigma_1 - \sigma_2) < \deg(b) - \deg(b)$. Also muss $\sigma_1 - \sigma_2 = 0$ gelten. Es folgt sofort, dass auch $\tau_1 = \tau_2$ und damit Widerspruch.

Wozu das? Partialbruchzerlegung zur Integration rationaler Funktionen.

$$\frac{c(x)}{a(x)b(x)} = \frac{\tau(x)}{a(x)} + \frac{\sigma(x)}{b(x)} \text{ mit } \deg(\tau) < \deg(a) \text{ und } \deg(\sigma) < \deg(b)$$

c) Wenn $a^i = b^j$ folgt aus der Eindeutigkeit der Primfaktorzerlegungen, dass es ein r gibt mit $a = r^k$ und $b = r^l$. Mit $a > b$ folgt das notwendige Kriterium $b|a$. Wenn a von a/b ersetzt wird, dann kann man aus der Lösung dieses Problems eine Lösung des ursprünglichen Problems bestimmen, indem der Exponent von b um den von a erhöht wird, und der von a/b für a genommen wird. Daraus folgt die Korrektheit.

Der Algorithmus berechnet im wesentlichen die größte gemeinsame Wurzel r von a und b , d.h. es gilt $\text{ggT}(k, l) = 1$. Für i und j müsste man $v := \text{kgV}(k, l) = kl$ bestimmen und könnte dann $i = v/l, j := v/k$ setzen. Der Algorithmus müsste dann auch noch so modifiziert werden, dass k, l berechnet werden.

Die Termination folgt daraus, dass das Produkt ab in jedem Schritt in ein Produkt $a'bb$ zerlegt wird, und im nächsten Schritt das Produkt $a'b$ vorliegt. Da $b > 1$ bis zur Terminierung gelten muss und $a \geq b$, wird ab in jedem Schritt wenigstens halbiert.

Die Komplexität: Es gilt $\log(ab) \in O(\log a)$. Da in jedem Schritt mindestens halbiert wird, ist das eine Schranke für die Anzahl der Schritte. In jedem Schritt wird a dividiert, was in $O((\log a)^2)$ möglich ist. Das ergibt zusammen $O((\log a)^3)$ als obere Schranke.

Es geht genauer: Wir nehmen an, dass zur Berechnung von a/b im wesentlichen $\log(a/b) \log(b)$ Bit-Operationen nötig sind. Sei $T(a, b)$ die Anzahl der Bit-Operationen

bei Eingabe a, b . Dann haben wir:

$$T(a, b) = \begin{cases} 0 & \text{falls } b = 1 \\ T(a/b, b) + \log(a/b) \log(b) & \text{falls } a \geq b^2, b > 1 \\ T(b, a/b) + \log(a/b) \log(b) & \text{falls } a < b^2, b > 1 \end{cases}$$

Und es wird gezeigt, dass $T(a, b) \leq \frac{1}{2}(\log a)^2$

Induktionsanfang: Die Eingaben $(a, 1)$ für alle a erfordern keine Division. Die kleinste Eingabe bei der dividiert wird, ist $(3, 2)$, dann haben wir $\log(3/2) \log 2 = (\log 3 - \log 2) \log 2 < (\log 3)^2 - (\log 2)^2 < \frac{1}{2}(\log 3)^2$

Induktionsschritt: Im ersten Fall: $(T(a, b) = \frac{1}{2} \log(a/b)^2 + \log(a/b) \log(b) = \frac{1}{2}(\log(a) - \log(b))^2 + (\log(a) - \log(b)) \log(b) = \frac{1}{2}(\log(a))^2 - \frac{1}{2}(\log(b))^2 < \frac{1}{2}(\log(a))^2$.

Im zweiten Fall: $T(a, b) = T(b, a/b) + \log(a/b) \log(b) = \frac{1}{2}(\log(a))^2 - \frac{1}{2}(\log a - \log b)(\log a + \log b) + (\log a - \log b) \log b = \frac{1}{2}(\log(a))^2 - \frac{1}{2}(\log a - \log b)^2 \leq \frac{1}{2}(\log(a))^2$.

zu **Aufgabe6:**

a) Zeige: $\hat{q} \geq q$.

Da $\lfloor a/b \rfloor < b$, gilt die Behauptung für $\hat{q} = b - 1$. Also sei $\hat{q} < b - 1$. Aus der Definition von \hat{q} folgt, dass $\hat{q}v_1 \geq u_0b + u_1 - v_1 + 1$. Also gilt:

$$u - \hat{q}v \leq u - \hat{q}v_1b^{n-1} \leq u_0b^n + \dots + u_n \quad (1)$$

$$- (u_0b^n + u_1b^{n-1} - v_1b^{n-1} + b^{n-1}) \quad (2)$$

$$= u_2b^{n-2} + \dots + u_n - b^{n-1} + v_1b^{n-1} \quad (3)$$

$$< v_1b^{n-1} \leq v \quad (4)$$

Da $u - \hat{q}v < v$ muss $\hat{q} \geq q$ gelten. □

Zeige: Für $v_1 \geq \lfloor b/2 \rfloor$ gilt $\hat{q} - 2 \leq q$.

Beweis: Es gilt

$$\hat{q} \leq \frac{u_0b + u_1}{v_1} = \frac{u_0b^n + u_1b^{n-1}}{v_1b^{n-1}} \leq \frac{u}{v_1b^{n-1}} < \frac{u}{v - b^{n-1}}.$$

Im Fall $v = b^{n-1}$ gilt $\hat{q} = q$. Sei $v \neq b^{n-1}$. Angenommen $\hat{q} \geq q + 3$, dann ist

$$3 \leq \hat{q} - q < \frac{u}{v - b^{n-1}} - \frac{u}{v} + 1 = \frac{u}{v} \cdot \frac{b^{n-1}}{v - b^{n-1}} + 1$$

Damit gilt

$$\frac{u}{v} > 2 \left(\frac{v - b^{n-1}}{b^{n-1}} \right) \geq 2(v_1 - 1).$$

Es gilt aber auch $b - 4 \geq \hat{q} - 3 \geq q \geq 2(v_1 - 1)$. Daraus folgt $v_1 < b/2$

b) 4100/588

Übungen zur Vorlesung Computeralgebra
Blatt 3

Prof. Dr. Klaus Madlener

7. Aufgabe: [Ringe]

- a) Ist für Integritätsbereiche R, S der Ring $R+S$ (die direkte Summe) ebenfalls immer ein Integritätsbereich? Zeigen oder widerlegen Sie.
- b) Sei $I = \{f \in \mathbb{R}[x] \mid f(5) = 0\}$ die Menge der reellen Polynome, die 5 als eine Nullstelle haben. Zeigen Sie, dass I ein Ideal in $\mathbb{R}[x]$ ist. Geben Sie einen Isomorphismus $\mathbb{R}[x]/I \rightarrow \mathbb{R}$ an.

8. Aufgabe:

- a) Bestimmen Sie in $\mathbb{Z}[[x]]$ das Inverse zur formalen Potenzreihe

$$a(x) = \sum_{k=0}^{\infty} a_k x^k = 1 + x + 2x^2 + 3x^3 + 5x^4 + \dots$$

mit $a_k = a_{k-1} + a_{k-2}$ für $k \geq 2$.

9. Aufgabe: [Simplifikation]

- a) Betrachten Sie die folgende „Definition“ einer *expandierten kanonischen Form* für multivariate Polynom-Ausdrücke über einem Integritätsbereich D :
1. Ausmultiplizieren aller Produkte von Polynomen
 2. Terme gleichen Grades zusammenfassen
 3. Terme nach fallenden Grad ordnen

Ist dies ein kanonischer Simplifikator gemäß der Definition aus der Vorlesung? Welche Angaben brauchen Sie, um diese Frage beantworten zu können? Ist die obige Definition eindeutig?

Wenden Sie die Vorschrift auf folgenden Ausdruck an:

$$a(x, y) = ((x^2 - xy + x) + (x^2 + 3) \cdot (x - y + 1)) \cdot ((y^3 - 3y^2 - 9y - 5) + x^4 \cdot (y^2 + 2y + 1))$$

- b) Finden Sie „einfachste“ Ausdrücke, die äquivalent zu den folgenden Ausdrücken sind:

$$- a(x, y) = \frac{1}{x^9 + x^8y + x^7y^2 + x^6y^3 + x^5y^4 + x^4y^5 + x^3y^6 + x^2y^7 + xy^8 + y^9}$$

$$- b(x, y) = \frac{x - 4}{x^5 + x^4y + x^3y^2 + x^2y^3 + xy^4 + y^5} - \frac{x^2 - xy + y^2}{x^6 - y^6}$$

- c) Wir betrachten ein freies Monoid $M = (\Sigma, \circ)$, wobei die Monoid-Elemente die Wörter über dem Alphabet Σ sind, das neutrale Element ϵ das leere Wort über dem Alphabet Σ ist und \circ eine assoziative Abbildung von Paaren auf Wörtern auf ein Wort ist mit $a \circ b = ab, a, b \in \Sigma^*$. Für ein $w \in M$ sei $(w)^n, n \in \mathbb{N}_+$ definiert durch $(w)^1 = w$ und $(w)^n = w(w)^{n-1}$. Überlegen Sie sich, wie hier Normalformen minimaler Länge von Wörtern aus M aussehen, d.h. es sollen Terme mit möglichst wenige Symbolen gefunden werden, die dasselbe Wort darstellen. Können Sie einen effizienten Algorithmus angeben, der diese Normalformen berechnet?

10. Aufgabe: [Potenzen]

Sei $n \in \mathbb{N}^+$. Eine Additions-kette für n ist eine endliche Folge positiver ganzer Zahlen a_0, \dots, a_r mit $a_0 = 1, a_r = n$, und für alle $i = 1, \dots, r$ gibt es j und k mit $k \leq j < i$, so dass $a_i = a_j + a_k$; r bezeichnen wir als die Länge der Additions-kette. Es bezeichne weiter $l(n)$ die minimale Länge einer Additions-kette für n .

- a) Welcher Zusammenhang besteht zwischen $l(n)$ und der Berechnung von x^n ? Ist die Anzahl der Multiplikationen der binären Potenzierungsmethode zur Berechnung von x^n immer minimal?
- b) Sei $s(n)$ die Summe der Bits in der Binärdarstellung von n (also die Anzahl der Einsen). Zeigen Sie: $l(n) \leq \lfloor \log_2 n \rfloor + s(n) - 1$ und $l(mn) \leq l(m) + l(n)$.
- c) Seien $a > b \geq 0$ ganze Zahlen. Zeigen Sie, dass $l(2^a) = a$ und $l(2^a + 2^b) = a + 1$.

11. Aufgabe: [Fibonacci]

- a) Wir wollen die Fibonacci-Zahlen F_k modulo $n \in \mathbb{N}$ berechnen. Geben Sie eine asymptotische obere Schranke für die Anzahl der Bitoperationen eines naiven Verfahrens zur Berechnung von F_k modulo n im schlechtesten Fall an.

Ein anderes Verfahren benutzt die Tatsache, dass F_{k+1} sich wie folgt gewinnen lässt:

$$\begin{pmatrix} F_{k+1} & F_k \\ F_k & F_{k-1} \end{pmatrix} = \begin{pmatrix} F_k & F_{k-1} \\ F_{k-1} & F_{k-2} \end{pmatrix} \cdot X,$$

wobei X von Ihnen anzugeben ist. Konkretisieren Sie nun dieses Verfahren zur Berechnung von F_k modulo n und geben Sie ebenfalls eine asymptotische obere Schranke für die Anzahl der Bitoperationen im schlechtesten Fall an.

- b) Zeigen Sie, wie man $1 + x + \dots + x^{n-1} \bmod m$ für $x, n, m \in \mathbb{N}^+$ mit $O(\lambda(n)\lambda(m)^2)$ Bitoperationen im schlechtesten Fall berechnet. Beachten Sie besonders den Fall, dass m keine Primzahl ist.

zu **Aufgabe7:**

- a) $(0, 1) \cdot (1, 0) = (0, 0)$ also kein Integritätsbereich
b) $I = \{f \mid f = q(x - 5)\}$, denn alle Polynome mit Nullstelle 5 haben $x - 5$ als Teiler. I ist ein Ideal, weil abgeschlossen bezüglich $+$ und \cdot . Betrachtet man die Abbildung $F : \mathbb{R}[x] \rightarrow \mathbb{R}[x]/I$, mit $F(f) = r + I$, wobei q und r durch $f = q(x - 5) + r$ im euklidischen Ring $\mathbb{R}[x]$ eindeutig definiert sind, dann gilt $\nu(r) < 1 = \nu(x - 5)$ oder $r = 0$, also $r \in \mathbb{R}$. Es folgt sofort die Surjektivität der Abbildung $r + I \rightarrow r$. Die Injektivität folgt aus der Gradbedingung von r , nur 0 ist sowohl in I als auch in \mathbb{R} . Sie ist auch ein Homomorphismus, also ein Isomorphismus.

zu **Aufgabe8:**

$$\begin{aligned} a_0 = 1 \text{ ist Einheit} & & b_0 = 1 \\ b_1 = -a_0^{-1}(a_1 b_0) = -1(1 \cdot 1) = -1 & & b_2 = -a_0^{-1}(a_1 b_1 + a_2 b_0) = -1(1 \cdot (-1) + 2 \cdot 1) = -1 \\ b_i = 0, i \geq 3 & & \end{aligned}$$

Also ist das formale Inverse zu $a(x) b(x) = 1 - x - x^2$.

zu **Aufgabe9:**

- a) Der Simplifikator liefert keine eindeutigen Normalformen. Man müsste die Äquivalenz so definieren, dass zwei Listen von Monomen dann äquivalent sind, wenn die Monome in der einen Liste sich nur in der Reihenfolge von denen der anderen Liste unterscheiden.

Was bedeutet „abfallende Grade“ bei multivariaten Polynomen? Es gibt viele Möglichkeiten Monome anzuordnen, das wird bei Gröbnerbasen noch eine wichtige Rolle spielen. Man kann Grade bezüglich einzelner Variablen betrachten, man kann den Totalgrad definieren. Dann könnte man die Simplifikationsordnung total bezüglich jeder Menge äquivalenter Terme machen. Wenn der Simplifikator kanonisch ist, bekommt man in einem Schritt die Normalform, wenn nicht kann man ihn bei einer wohlfundierten Simplifikationsordnung kanonisch machen, indem man ihn solange wiederholt, bis nach endlich vielen Schritten ein Fixpunkt erreicht ist.

Welcher Simplifikator macht denn überhaupt Sinn, für polynomiale Ausdrücke? Im wesentlichen keiner. Faktorisierung bringt oft die kürzesten Ergebnisse, und bei rationalen Ausdrücken auch die besten Chancen zum Kürzen, aber Faktorisierung von Polynomen ist teuer, zwar polynomial, aber zu teuer, um sie ständig anzuwenden.

- b) Der Schlüssel ist hier zu faktorisieren. $(x^n - y^n)/(x - y) = x^{n-1} + x^{n-2}y + \dots + xy^{n-2} + y^{n-1}$

c) Im wesentlichen ist man bei diesem Problem schon mitten in Kompressionsalgorithmen. Um die Lesbarkeit zu erhalten muss man auf die heute üblichen Algorithmen, die auf Wörterbüchern basieren, verzichten. Eines der ältesten Verfahren, das aber heute als Vorstufe zu anderen Algorithmen noch in Gebrauch ist, ist Run-Length-Encoding, die aber üblicherweise nur Potenzen von einem Buchstaben betrachtet und auch schnell berechnet. Wenn man zu Potenzen von längeren Teilwörtern geht, wird es schnell schwieriger. Es ist möglich, dass sich solche Potenzen überlappen, beispielsweise *abababcbbc*. Also hat man hier Lösungen für ein Optimierungsproblem zu suchen, die potentiell exponentiell viele verschiedene Möglichkeiten bieten. Selbst wenn sich ein Kriterium ergeben

sollte, die Anzahl polynomial zu beschränken, könnte hier dasselbe Problem auftreten, wie bei den Polynomen, die Simplifikation wird so teuer, dass sie nur auf spezielle Anforderung des Benutzers ausgeführt wird, z.B. um einen Bildschirm voll Daten für den Benutzer besser darzustellen.

zu **Aufgabe10:**

a) Eine Additionskette für n liefert ein Verfahren, eine Potenz a^n durch fortgesetzte Multiplikation mit einer gewissen Anzahl Zwischenregistern zu berechnen. Für die gewöhnliche binäre Exponentiation, braucht man nur 3 Register, die Basis, der Exponent, und das Ergebnis. Mit allgemeinen Additionsketten braucht man eventuell noch mehr Register für Zwischenergebnisse. Eine Additionskette für 15 ist beispielsweise 1,2,4,5,10,15, eine andere korrespondierend zur binären Potenzierungsmethode wäre 1,2,4,8,12,14,15, womit auch die Frage beantwortet wäre, ob diese immer optimal ist.

b) Die erste Ungleichung ist gerade das Ergebnis, das die binäre Methode liefert, zuerst $\lceil \log_2 n \rceil$ 2er-Potenzen, und für jedes 1-Bit außer dem ersten ein weiteres Zwischenergebnis.

Die zweite Ungleichung ergibt sich sofort daraus, dass man eine Additionskette für n mit m Multiplizieren kann und sie ohne das führende m an die für m anhängen kann, wobei dann eine neue Additionskette für mn entsteht. Eine optimale Additionskette für mn kann bestenfalls noch kürzer sein.

c) Wenn man immer den bisher größten Wert verdoppelt, erreicht man 2^a nach a Schritten, und es kann keine kürzere Kette geben. Mit demselben Argument erhält man $l(n) \geq \lceil \log_2 n \rceil$

$l(2^a + 2^b) \leq a + 1$ ist klar, die binäre Methode liefert $a + 1$. Es gilt $\lceil \log_2 2^a + 2^b \rceil = a + 1 \leq l(2^a + 2^b)$

Anmerkung: Siehe Knuth, Seminumerical Algorithms Kap. 4.6.3.

zu **Aufgabe11:**

a) Wir haben k Schritte mit je einer Addition modulo n zu berechnen, d.h. wir teilen in jedem Schritt mit Rest durch n , was man aber auch so realisieren kann, dass n abgezogen wird, wenn die Summe größer als n wird. Also haben wir $O(k \log_2(n))$ Bit-Operationen für das naive Verfahren.

Beim Verfahren mit den Matrizen, hat man eine Additionskette mit der Länge $2 \log_2(k) - 1$ im schlimmsten Fall, um die Potenz der Ausgangsmatrix zu bestimmen. Jedes Produkt von zwei Matrizen benötigt $O(\log_2(n)^2)$ Bitoperationen, für jeden der 4 Einträge im Produkt 2 Multiplikationen und eine Addition. Zusammen ergibt das $O(\log_2(k) \log_2(n)^2)$. Für genügend große k gegenüber n gewinnt also immer der Algorithmus mit den Matrizen.

b) Die naheliegenderere Methode ist $(x^n - 1)(x - 1)^{-1} \bmod m$ zu berechnen. Dazu muss aber $\text{ggT}(x - 1, m) = 1$ sein, damit $x - 1$ invertierbar modulo n ist. Man kann aber für $x \neq 1$ $r = (x^n - 1) \bmod m(x - 1)$ berechnen. Dann gilt $(x - 1) \mid r$ und $r/(x - 1)$ ist das gesuchte Ergebnis. Für $x = 1$ ist $n \bmod m$ die gesuchte Lösung.

Zur Exponentiation werden $O(\log_2 n)$ Schritte gebraucht, jedesmal werden $O((\log_2 x + \log_2 m)^2)$ Bitoperationen gebraucht. Es verbleibt dann noch eine Division in $O((\log_2 m)^2)$

Bitoperationen. Zusammen ergibt sich das gewünschte Ergebnis, wenn $x \in O(m)$.

Ein anderes Verfahren benutzt wieder Matrizen, diesmal die folgende

$$M = \begin{pmatrix} 1 & 1 \\ 0 & x \end{pmatrix}, M^i = \begin{pmatrix} 1 & 1 + x + \dots + x^i \\ 0 & x^{i+1} \end{pmatrix}, i \in \mathbb{N}$$

Für die Länge der Additions-kette von n gilt das oben gesagte. In jedem Schritt sind jetzt $O((\log_2 m)^2)$ Bitoperationen zu machen, genauer 4 Multiplikationen zu $(\log_2 m)^2$ und eine Addition zu $\log_2 m$ Bitoperationen. Zusammen ergibt sich die gewünschte Schranke, aber für beliebige x .

Übungen zur Vorlesung Computeralgebra
Blatt 4

Prof. Dr. Klaus Madlener

12. Aufgabe: [Potenzreihen]

Gegeben sei ein Körper F der Charakteristik 0. Zeigen Sie: Die Koeffizienten $a_k \in F$ für $k \geq K$ (K fest) einer Potenzreihe $a(x) = \sum_{k=0}^{\infty} a_k x^k$ können genau dann durch eine lineare Rekurrenzgleichung mit konstanten Koeffizienten $a_k = u_1 a_{k-1} + u_2 a_{k-2} + \dots + u_n a_{k-n}$ (n fest) über F dargestellt werden, wenn $a(x)$ als rationale Funktion in x über F dargestellt werden kann.

13. Aufgabe: [Pseudo-Restefolgen]

Untersuchen Sie, ob die polynomialen Pseudo-Restefolgen aus der Vorlesung tatsächlich zur ggT-Berechnung verwendet werden können. Zeigen Sie dazu:

Sind $a(x)$ und $b(x)$ primitive Polynome über einem ZPE-Ring D und ist $f_1(x) = a(x)$, $f_2(x) = b(x)$, $f_3(x), \dots, f_{k-1}(x), f_k(x)$ eine polynomiale Restefolge für $a(x)$ und $b(x)$, wobei $f_k(x) = 0$ sei, so gilt:

$$\text{ggT}(a(x), b(x)) = \text{pp}(f_{k-1}(x)).$$

14. Aufgabe: [Anwendung]

Sei D ein euklidischer Ring mit Bewertungsfunktion ν . Entwerfen Sie einen Algorithmus, der der folgenden Spezifikation genügt:

Eingabe: Eine positive ganze Zahl n sowie $a, d_1, \dots, d_n \in D \setminus \{0\}$ mit $\text{ggT}(d_i, d_j) = 1$ für $i \neq j$.

Ausgabe: $a_0, a_1, \dots, a_n \in D$, so dass

$$\frac{a}{d_1 \cdots d_n} = a_0 + \sum_{i=1}^n \frac{a_i}{d_i}$$

und entweder $a_i = 0$ oder $\nu(a_i) < \nu(d_i)$ für $i \geq 1$.

Begründen Sie auch, weshalb Ihr Algorithmus korrekt ist. Ist Ihnen dieser Algorithmus schon einmal begegnet?

15. Aufgabe: [Nachrichtenverifikation]

Nehmen Sie an, Alice und Bob hätten jeweils eine Nachricht von n Bit, M_A und M_B . Sie würden gerne verifizieren, dass ihre Nachrichten identisch sind, und zwar einerseits mit einer großen Wahrscheinlichkeit und andererseits mit wenig Kommunikationsaufwand. Insbesondere sei n so groß, dass ein Nachrichtenaustausch ausscheide.

Alice und Bob wählen k Primzahlen p_1, \dots, p_k uniform aus der Menge der ersten $2n$ Primzahlen und prüfen dann, ob $M_A \equiv M_B \pmod{p_i}$ für $1 \leq i \leq k$. Zeigen Sie, dass, wenn $M_A = M_B$, so ist $M_A \equiv M_B \pmod{p_i}$ für alle i , während, wenn $M_A \neq M_B$, so ist $M_A \not\equiv M_B \pmod{p_i}$ für ein i mit Wahrscheinlichkeit mindestens $1 - 2^{-k}$.

16. Aufgabe: [Karatsuba]

- a) Machen Sie sich klar, dass der Multiplikationsalgorithmus nach Karatsuba und Ofman (in der Vorlesung für Langzahlarithmetik eingeführt) auch für univariate Polynome funktioniert. Formulieren Sie eine entsprechende rekursive Prozedur.
- b) Zeigen Sie, dass dieser Algorithmus eine Multiplikation zweier Polynome vom Grade höchstens n (wobei n eine Zweierpotenz sei) mit höchstens $9n^{\log_2 3} + O(n)$ Ringoperationen durchführt.

Zeigen Sie dazu folgendes Lemma:

Seien $b, d \in \mathbb{N}$ mit $b > 0$, und seien $S, T : \mathbb{N} \rightarrow \mathbb{N}$ Funktionen mit $S(2n) \geq 2S(n)$ sowie $S(n) \geq n$ für alle $n \in \mathbb{N}$. Es gelte $T(1) = d$ und $T(n) \leq bT(n/2) + S(n)$ für $n = 2^i$ und $i \in \mathbb{N}^+$. Dann gilt für $i \in \mathbb{N}$ und $n = 2^i$:

$$T(n) \leq \begin{cases} (2 - 2/n)S(n) + d \in O(S(n)) & \text{falls } b = 1, \\ S(n) \log_2 n + dn \in O(S(n) \log_2 n) & \text{falls } b = 2, \\ \frac{2}{b-2}(n^{\log_2 b-1} - 1)S(n) + dn^{\log_2 b} \in O(S(n)n^{\log_2 b-1}) & \text{falls } b \geq 3. \end{cases}$$

- c) überzeugen Sie sich, dass der Algorithmus nach Karatsuba und Ofman für „kleine“ Eingaben langsamer ist als der klassische Multiplikationsalgorithmus für Polynome.

Untersuchen Sie nun einen hybriden Algorithmus, der rekursiv die Karatsuba/Ofman-Idee anwendet, bis die Grade kleiner als eine Grenze $2^d \in \mathbb{N}$ werden, und dann die klassische Multiplikation verwendet.

Zeigen Sie, dass dieser hybride Algorithmus höchstens $\gamma(d)n^{\log_2 3} + O(n)$ Ringoperationen durchführt, wobei $\gamma(d)$ nur von d abhängt. Finden Sie d , so dass $\gamma(d)$ minimal ist. (Dazu brauchen Sie eine recht genaue Abschätzung für die Anzahl der Ringoperationen im klassischen Multiplikationsalgorithmus.)

zu **Aufgabe13:**

Mann kann eine Ähnlichkeitsrelation \sim definieren durch $a(x) \sim b(x)$, gdw. es gibt $\alpha, \beta \in D$ mit $\alpha a = \beta b$.

Eine polynomiale Restefolge (f_i) liegt vor, wenn $\deg(f_1) > \deg(f_2)$, nur das letzte Folgenglied 0 ist, und $f_i \sim \text{prem}(f_{i-1}f_{i-2})$.

Wir hatten in der Vorlesung Satz 4.4, der besagt, dass $\text{ggT}(a(x), b(x)) = \text{ggT}(b(x), PP(r(x)))$, wobei $r(x)$ der Rest der Pseudodivision von $a(x)$ und $b(x)$ ist.

Also gilt: $\text{ggT}(f_1(x), f_2(x)) \sim \dots \sim \text{ggT}(f_{k-1}(x), f_k(x)) \sim f_{k-1}(x)$. Mit dem Gaußschen Lemma folgt, dass $\text{pp}(f_{k-1}(x)) = \text{ggT}(a(x), b(x))$

zu **Aufgabe14:**

Sei $d = \prod d_i$. Berechne $a = da_0 + r$ mit euklidischem Algorithmus. Berechne $r = a_1(d_2 \cdots d_n) + r_1 d_1$, geht weil $\text{ggT}(d_k \cdots d_n, d_{k-1}) = 1$. Also haben wir: $a/d = a_0 + r/d = a_0 + a_1/d_1 + r/(d_2 \cdots d_n)$. Eine Zerlegung für $r/(d_2 \cdots d_n)$ läßt sich rekursiv gewinnen, es gibt also r_0, a_2, \dots, a_n mit $r/(d_2 \cdots d_n) = r_0 + \sum_{i=2}^n a_i/d_i$. Damit hat man die gesuchte Zerlegung. Da die d_i teilerfremd sind existieren die ggTs. Der Abbruch der Rekursion für den Fall $d = d_1$ ist ebenfalls klar, dann erhält man $a/d = a_0 + a_1/d$. Der Algorithmus liefert eine Partialbruchzerlegung rationaler Funktionen, oft erster Schritt bei der Integration. Die Faktorisierung ist für die Polstellen unerlässlich, mit der Gradeigenschaft hat man oft, dass die Summanden eine einfach erkennbare Stammfunktion haben, wenn man zusätzlich noch annimmt, dass die d_i höchstens Grad 2 haben für Polynome über \mathbb{R} .

zu **Aufgabe15:**

Wenn $M_A = M_B$, dann gilt $M_A \equiv M_B \pmod{p_i}$. Wir zeigen, wenn $M_A \neq M_B$, dann gilt $M_A \not\equiv M_B \pmod{p_i}$ mit Wahrscheinlichkeit $t < 1/2$. Sei P die Menge der ersten $2n$ Primzahlen, $S = \{p \in P : M_A \equiv M_B \pmod{p}\}$. Angenommen $|S| \geq n$. Dann wäre $M_A \equiv M_B \pmod{g}$, wobei $g = \prod_{s \in S} s \geq 2^{|S|} \geq 2^n$. Dann wäre aber $M_A = M_B$. Also muss $|S| < n$ gelten. Setze $t = |S|/|P| < 1/2$. Wenn k Primzahlen uniform aus P gewählt werden, hat man die Wahrscheinlichkeit von höchstens 2^{-k} , dass alle Paare von Moduli sich entsprechen, obwohl die Nachrichten verschieden sind.

Bei den Abgaben wurde zum Teil so vorgegangen, dass k beliebige Primzahlen gewählt wurden und das Produkt dieser Primzahlen durch 2^k abgeschätzt wurde. Dabei tritt das Problem auf, dass man im wesentlichen keine Aussage über die Wahrscheinlichkeit hat. Man muss schon sagen, aus welchem Bereich man die Primzahlen wählt, erst dann hat man eine Wahrscheinlichkeit für eine korrekte Aussage, bevor so viele Primzahlen gewählt wurden, dass im wesentlichen die ganze Information übertragen wurde.

zu **Aufgabe16:**

- Zerlegung modulo $x^{n/2}$, rekursive Aufrufe, Zusammensetzen.
- Durch schrittweises Auflösen der Rekursion erhält man

$$T(2^i) \leq bT(2^{i-1}) + S(2^i) \leq b(bT(2^{i-2}) + S(2^{i-1})) + S(2^i) \quad (5)$$

$$= b^2T(2^{i-2}) + bS(2^{i-1}) + S(2^i) \leq \dots \quad (6)$$

$$\leq b^i T(1) + \sum_{0 \leq j < i} b^j S(2^{i-j}) \leq d 2^{i \log b} + S(2^i) \sum_{0 \leq j < i} \left(\frac{b}{2}\right)^j, \quad (7)$$

wobei im letzten Schritt $S(2^{i-j}) < 2^{-j}S(i)$ ausgenutzt wurde. Für $b = 2$ vereinfacht sich die letzte Summe zu $S(2^i) \cdot i$. Wenn $b \neq 2$ bleibt die geometrische Summe

$$\sum_{0 \leq j < i} \left(\frac{b}{2}\right)^j = \frac{\left(\frac{b}{2}\right)^i - 1}{\frac{b}{2} - 1} = \frac{2}{b-2} (2^{i(\log b-1)} - 1)$$

Daraus ergeben sich die gesuchten Abschätzungen für das Lemma.

Die Berechnungen Summen für den dritten rekursiven Aufruf benötigen n Ringadditionen. Beim Zusammensetzen des Ergebnisses hat man nur für den mittleren Teil $2n$ Subtraktionen und weitere n Additionen, um dessen Koeffizienten zu denen des vorderen und hinteren Teils zu addieren. Also kann man $b = 3$, $d = 1$ und $S(n) = 4n$ setzen und das Lemma anwenden mit dem Ergebnis $9n^{\log 3} - 8n$ Koeffizientenoperationen.

c) Für die klassische Multiplikation haben wir $(n+1)^2$ Multiplikationen und $(n+1)^2 - 2(n+1) + 1$ Additionen, also zusammen $2n^2 + 2n + 1$ Operationen. Die Gleichung $2n^2 + 2n + 1 - (9n^{1.59} - 8n) = 0$ ergibt als Lösungen 2.47 und rund 25.14, also ein Break-Even von etwa $2^{4.65}$.

Wir erhalten für den hybriden Algorithmus mit Grenze 2^e , indem wir die Rekursion nur bis dahin ausschreiben, und dann für Polynome mit Grad höchstens 2^e den Aufwand durch die klassische Methode mit $T(2^e) = 2(2^e + 1)^2 - 22^e + 1 = 22^{2^e} + 22^e + 1$ ansetzen:

$$T(2^i) \leq bT(2^{i-1}) + S(2^i) \leq b(bT(2^{i-2}) + S(2^{i-1})) + S(2^i) \quad (8)$$

$$= b^2T(2^{i-2}) + bS(2^{i-1}) + S(2^i) \leq \dots \quad (9)$$

$$\leq b^{i-e}T(2^e) + \sum_{0 \leq j < i-e} b^j S(2^{i-j}) \leq b^{i-e}T(2^e) + S(2^i) \frac{2}{b-2} (2^{(i-e)(\log_2 b-1)} - 1) \quad (10)$$

$$\leq b^{i-e} \left(T(2^e) + \frac{2}{b-2} 2^e \right) - S(2^i) \frac{2}{b-2} \quad (11)$$

$$= 3^i \frac{2 \cdot 2^{2^e} + 4 \cdot 2^e + 1}{3^e} - 2S(2^i) \quad (12)$$

Wir setzen $\gamma(e) = \frac{2 \cdot 2^{2^e} + 4 \cdot 2^e + 1}{3^e}$. γ ist für $e = 2.43$ minimal mit einem Wert von 5.389. $\gamma(2) > \gamma(3) \approx 5.4$, also erreichen wir mit $e = 3$ eine Verbesserung der Konstante von 9 auf 5.4.

Dieses Ergebnis wird aber mit exakt diesen Werten nicht in der Realität zu erreichen sein, wir haben nur die Operationen im Koeffizientenring gezählt, Speicherallokationen und einige andere Einflussfaktoren, die eher zu Lasten des Karatsuba-Algorithmus gehen, sind nicht berücksichtigt worden.

Übungen zur Vorlesung Computeralgebra
Blatt 5

Prof. Dr. Klaus Madlener

17. Aufgabe:

Entwickeln Sie Möglichkeiten, um diophantische Gleichungen auch in nicht-euklidischen Bereichen zu lösen. Betrachten Sie

1. $ax + by \equiv c \pmod{n}$, wobei $n \in \mathbb{N}$.
2. Beliebige diophantische Gleichungen über $\mathbb{Z}[x]$. (Vorsicht!)

18. Aufgabe:

Nehmen Sie an, dass ungerade und teilerfremde Moduli m_j gegeben seien. Außerdem sei $u = (u_1, \dots, u_r)$ als Restvektor bezüglich der m_j dargestellt. Wie kann man (unter der Annahme, dass u ein Vielfaches von 2 ist) $u/2$ modular berechnen?

19. Aufgabe:

- a) Sei F ein Körper, $f(x)$ ein univariates Polynom über F und $u \in F$. Geben Sie eine Methode zur Berechnung von $f(u)$ an, die mit möglichst wenigen Operationen in F auskommt.
- b) Verallgemeinern Sie diese Methode auf Polynome aus $F[x, y]$ und geben Sie auch hier Abschätzungen für die Anzahl der Körperoperationen an.

20. Aufgabe:

- a) Geben Sie für $n = 4$ und $n = 8$ jeweils primitive Einheitswurzeln in \mathbb{C} an und berechnen Sie die Fouriertransformierten von $(0, 1, 2, 3)$ und $(1, 2, 0, 2, 0, 0, 0, 1)$.
- b) Seien $a(x) = -x^3 + 3x + 1$ und $b(x) = 2x^4 - 3x^3 - 2x^2 + x + 1$ Polynome aus $\mathbb{Z}_{17}[x]$. Bestimmen Sie das Produkt dieser Polynome mit Hilfe der schnellen Fourier-Transformation.

21. Aufgabe:

Sei $a(x)$ ein Polynom vom Grade $3^n - 1$ mit $n \geq 0$.

- a) Zeigen Sie, dass $a(x)$ als

$$a(x) = b(x^3) + x \cdot c(x^3) + x^2 \cdot d(x^3)$$

geschrieben werden kann; dabei sind $b(x)$, $c(x)$ und $d(x)$ Polynome vom Grade kleiner oder gleich $3^{n-1} - 1$.

- b) Finden Sie Symmetriebedingungen ähnlich zu denen aus der Vorlesung, die es erlauben, $a(x)$ an 3^n Stellen auszuwerten, indem man drei geeignete Polynome jeweils an 3^{n-1} geeigneten Stellen auswertet.
- c) Zeigen Sie, dass für eine primitive 3^n -te Einheitswurzel ω eines Körpers die Elemente $1, \omega, \dots, \omega^{3^n-1}$ die Symmetriebedingungen aus b) erfüllen.
- d) Welches sind die Kosten der Auswertung von $a(x)$ an den 3^n in c) genannten Stellen?
- e) Verwenden Sie a) bis d), um einen 3-FFT-Algorithmus zu entwickeln und eine Abschätzung der Anzahl der Multiplikationen im Koeffizientenkörper anzugeben.

zu **Aufgabe17:**

1. CRT+Newton-Iteration, Reduktion auf n Primzahl.
2. ggT geht, aber Hilbert's 10. Problem.

zu **Aufgabe18:**

Angenommen wir haben die modulare Darstellung einer Zahl $u = (u_1, \dots, u_n)$ bezüglich der Moduli m_1, \dots, m_n . Betrachtet man nun die Zahl $2u$, so ist deren modulare Darstellung $(2u_1 \pmod{m_1}, \dots, 2u_n \pmod{m_n})$. Da alle m_i ungerade sind, gibt es jeweils ein multiplikatives Inverses r_i von 2, bestimme per EEA $r_i 2 + tm_i = 1$. Also lässt sich die modulare Darstellung von u schreiben als $2u_1 r_1 \pmod{m_1}, \dots, 2u_n r_n \pmod{m_n}$

Wie könnte man bei einer ungeraden Zahl $\lfloor u/2 \rfloor$ bestimmen, natürlich auch modular?

zu **Aufgabe19:**

Horner-Schema: Anders klammern, um die Anzahl der Multiplikationen zu verringern. $a_n x^n + a_{n-1} x^{n-1} + a_{n-1} x^{n-1} + a_0 = (\dots(((a_n x) + a_{n-1})x + a_{n-2})x + \dots) + a_0$. Für ein Polynom vom Grad n hat man n Multiplikationen und n Additionen. Weitere Verbesserungen sind für spezielle Problemstellungen möglich.

Im zweidimensionalen Fall kann man die Polynome aus $F[x, y]$ als solche aus $F[y][x]$ betrachten, jede Potenz von x hat ein Polynom aus $F[y]$ als Koeffizient. Sei n der maximale Grad in x und m der maximale Grad in y , dann sind n Koeffizienten aus $F[y]$ zu berechnen, jeweils mit m Multiplikationen und m Additionen. Für die Auswertung der Polynome in x sind dann n Multiplikationen und n Additionen nötig, so dass zusammen $2n + 2mn$ Operationen nötig sind.

zu **Aufgabe20:**

a) Für $n = 4$ sind möglich $i = e^{\frac{2\pi i}{4}}$ oder $-i = e^{\frac{2\pi i \cdot 3}{4}}$. Die beiden anderen vierten Einheitswurzeln 1 und -1 sind nicht primitiv. Wir nehmen i . Die Vandermonde-Matrix zu i ist:

$$\text{VDM}(1, i, i^2, i^3) = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{pmatrix}$$

Die Fouriertransformierte von $v = (0, 1, 2, 3)$ ist dann $\text{VDM}(1, i, i^2, i^3) \cdot v = (6, -2 - 2i, -2, -2 + 2i)$

Für $n = 8$ sind möglich $e^{\frac{2\pi i j}{8}}$, wobei $j \in \{1, 3, 5, 7\}$. Für $j \in \{0, 2, 4, 6\}$ ist die Einheitswurzel nicht primitiv. Wir nehmen $\omega = e^{\frac{2\pi i}{8}}$. Rest geht analog.

b) \mathbb{Z}_{17} enthält primitive achte Einheitswurzeln, weil $8 \mid 17 - 1$ und $\text{ggT}(8, 17) = 1$, es ist sogar ein Körper, d.h. jedes Element $\neq 0$ ist Einheit. Es gilt $2^8 = 255 + 1 = 17 \cdot 15 + 1 \equiv 1 \pmod{17}$. Also ist 2 eine solche achte Einheitswurzel. Die Symmetriebedingung gilt, weil $2^4 \equiv 16 \equiv -1 \pmod{17}$ und damit $2^{i+n/2} = 2^{i+4} = -2^i$. 2 ist auch primitive Einheitswurzel, weil \mathbb{Z}_{17} ein Körper ist, und $\sum_{0 \leq j < n} 2^{pj} = 0$. Wir wenden den Algorithmus von Folie 139 an.

$$a(x) = -x^3 + 3x + 1 \text{ und } b(x) = 2x^4 - 3x^3 - 2x^2 + x + 1$$

FFT mit 2^1 (1, 3, 0, -1, 0, 0, 0, 0)
 FFT mit 2^2 (1, 0, 0, 0) (3, -1, 0, 0)
 FFT mit 2^4 (1, 0) (0, 0) (3, 0) (-1, 0)
 FFT mit 2^8 (1) (0) (0) (0) (3) (0) (-1) (0)
 Zusammensetzen mit 2^4 (1, 1) (0, 0) (3, 3) (-1, -1)
 Zusammensetzen mit 2^2 (1, 1, 1, 1) (2, -1, 4, 7)
 Zusammensetzen mit 2^1 3, -1, 0, 6, -1, 3, 2, -4

FFT mit 2^1 (1, 1, -2, -3, 2, 0, 0, 0)
 FFT mit 2^2 (1, -2, 2, 0) (1, -3, 0, 0)
 FFT mit 2^4 (1, 2) (-2, 0) (1, 0) (-3, 0)
 FFT mit 2^8 (1) (2) (-2) (0) (1) (0) (-3) (0)
 Zusammensetzen mit 2^4 (3, -1) (-2, -2) (1, 1) (-3, -3)
 Zusammensetzen mit 2^2 (1, 8, 5, 7) (-2, 6, 4, -4)
 Zusammensetzen mit 2^1 -1, 3, 4, -8, 3, -4, 6, 5

Produkt in Wertedarstellung: (-3, -3, 0, 3, -3, 5, -5, -3)

FFT mit $9^1 = 2^{-1}$ (-3, -3, 0, 3, -3, 5, -5, -3)
 FFT mit 9^2 (-3, 0, -3, -5) (-3, 3, 5, -3)
 FFT mit 9^4 (-3, -3) (0, -5) (-3, 5) (3, -3)
 FFT mit 9^8 (-3) (-3) (0) (-5) (-3) (5) (3) (-3)
 Zusammensetzen mit $9^4 \equiv 16 \equiv -1$ (-6, 0) (-5, 5) (2, -8) (0, 6)
 Zusammensetzen mit $9^2 \equiv -4$ (6, -3, -1, 3) (2, 2, 2, -1)
 Zusammensetzen mit 9^1 (8, -2, 8, 5, 4, -4, 7, 1)

Skalarmult. mit $8^{-1} \bmod 17 = 15$

Ergebnis: $1 + 4x + x^2 + 7x^3 - 8x^4 + 8x^5 + 3x^6 - 2x^7$

zu **Aufgabe21:**

Wozu wird 3-FFT benötigt? Angenommen wir wollen Polynome in $F_q[x]$ bearbeiten mit $q = 2^r$, also Polynome über einem Körper der Charakteristik 2. Es gibt aber wegen Satz 4.22 nur n -te primitive Einheitswurzeln, wenn $n|2^r - 1$, also insbesondere keine 2^s -ten Einheitswurzeln in $F_q[x]$ für ein s . Aber auch damit hat man nur eine eingeschränkte Wahl. Hier eine Tabelle mit Werten n und r minimal, so dass eine primitive $n = 3^k$ -te Einheitswurzel existiert:

k	1	2	3	4	5	6	7	8	9	10
r	2	6	18	54	162	486	1458	4374	13122	39366

Das bedeutet, um bei Polynomen vom Grad 10000 über Körpern der Charakteristik 2 3-FFT nutzen zu können, muss man mindestens in $F_{2^{13122}}[x]$ arbeiten, weil $k = \lceil \log_2(10000) \rceil = 9$.

a)

$$\sum_{i=0}^{3^n-1} a_i x^i = \sum_{i=0}^{3^{(n-1)}-1} a_{3i} x^{3i} + a_{3i+1} x^{3i+1} + a_{3i+2} x^{3i+2} = b(x) + xc(x) + x^2 d(x)$$

b) Eine Bedingung: $x_k^3 = x_{k+n/3}^3 = x_{k+2n/3}^3$. Wenn man dann die Zerlegung aus a) betrachtet, dann gelten mit $W = \omega^{n/3}$:

$$A_j = B_j + \omega^j C_j + \omega^{2j} D_j$$

$$A_{j+n/3} = B_j + \omega^{j+n/3} C_j + \omega^{2j+2n/3} D_j = B_j + W\omega^j C_j + W^2\omega^{2j} D_j$$

$$A_{j+2n/3} = B_j + \omega^{j+2n/3} C_j + \omega^{2j+4n/3} D_j = B_j + W^2\omega^j C_j + W^4\omega^{2j} D_j$$

wenn $B_j = b(\omega^j)$, $C_j = c(\omega^j)$ und $D_j = d(\omega^j)$ die rekursiv mit ω^3 als Einheitswurzel berechneten Auswertungen sind.

c) Für $l \in \{0, 1, 2\}$ gilt $(\omega^{k+ln/3})^3 = \omega^{3k+ln} = \omega^{3j}$. Das entspricht Lemma 4.15. Für die Korrektheit fehlt dann noch Lemma 4.16, so dass auch ω^3 für den nächsten rekursiven Schritt noch die Bedingungen erfüllt.

d) Wir haben 10 Multiplikationen und 6 Additionen der Koeffizienten pro 3 Stellen im Ergebnis. Das Zerlegen am Anfang erfordert maximal n Kopien.

e) Der Algorithmus ist analog zu dem in der Vorlesung. Wir zählen nur die Multiplikationen im Koeffizientenbereich. In jedem Schritt zwei zur Berechnung von W^2 und W^4 , zwei für A_j , und weitere vier für die Produkte mit den W , das sind zusammen $T(n) = 3T(n/3) + (10/3)n$, oder nach dem Master-Theorem $O(n \log n)$.

Übungen zur Vorlesung Computeralgebra
 Blatt 6

 Prof. Dr. Klaus Madlener

22. Aufgabe:

- a) Seien $f(x) = x^3 - x^2 + 2$ und $g(x) = x^2 + x + 1$ Polynome über \mathbb{Q} . Berechnen Sie eine Darstellung von $h(x) = x^4 + 2x$ als $h(x) = p(x)f(x) + q(x)g(x)$ mit $\deg p < 2$ und $\deg q < 3$.
- b) Berechnen Sie das Polynom $r(x) \in \mathbb{Q}[x]$ kleinsten Grades, das die Kongruenzen

$$\begin{aligned} r(x) &\equiv 2x^2 + 1 \pmod{x^3 + x^2 - 1} \\ r(x) &\equiv x + 2 \pmod{x^2 + 2x + 2} \end{aligned}$$

erfüllt.

23. Aufgabe:

Wir betrachten den Algorithmus von Garner aus der Vorlesung.

- a) Der zweite Schritt benutzt die Formeln

$$\begin{aligned} \nu_0 &\equiv u_0 \pmod{m_0} \\ \nu_k &\equiv \left(u_k - \left(\nu_0 + \nu_1 m_0 + \cdots + \nu_{k-1} \prod_{i=0}^{k-2} m_i \right) \right) \left(\prod_{i=0}^{k-1} m_i \right)^{-1} \\ &\pmod{m_k} \text{ für } k \geq 1. \end{aligned}$$

 Zeigen Sie, dass man die gemischten Basiskoeffizienten ν_k auch mit Hilfe der Formeln

$$\begin{aligned} \nu_0 &\equiv u_0 \pmod{m_0} \\ \nu_k &\equiv \left(\cdots \left((u_k - \nu_0)m_0^{-1} - \nu_1 \right) m_1^{-1} - \cdots - \nu_{k-1} \right) m_{k-1}^{-1} \\ &\pmod{m_k} \text{ für } k \geq 1 \end{aligned}$$

 berechnen kann. (Beachten Sie: Die Inversen in dieser Formel sind Inverse modulo m_k .)

- b) Wenn man den zweiten Schritt wie in a) realisiert, welche Menge von Inversen muss dann im ersten Schritt berechnet werden? Wie viele Inverse werden benötigt?

- c) Vergleichen Sie die Zeitkomplexität beider Varianten. Betrachten Sie einmal den Fall, dass die Menge $\{m_i\}$ der Reste fest ist, d. h. die Inversenbildung im ersten Schritt ein einen Vorberechnungsschritt ausgelagert werden kann, und auch den Fall, dass dies nicht möglich ist.

24. Aufgabe:

Bestimmen Sie mit Hilfe der 7-adischen linearen Newton-Iteration die dritte Wurzel des Polynoms

$$a(x) = x^6 - 531x^5 + 94137x^4 - 5598333x^3 + 4706850x^2 - 1327500x + 125000$$

mit $a(x) \in \mathbb{Z}[x]$. Führen Sie diese Rechnung mit Hilfe eines Computeralgebra-Systems durch.

25. Aufgabe:

- a) Es seien $I = \langle x, y \rangle$ und $J = \langle x \rangle$ Ideale in $\mathbb{Z}[x, y]$. Beschreiben Sie zunächst die Elemente von I und J sowie die Teilmengenbeziehung zwischen I und J . Beschreiben Sie dann die Elemente von $I + J$, $I \cdot J$ und I^2 und geben Sie erzeugende Elemente dieser Ideale an. Untersuchen Sie schließlich die Teilmengenbeziehungen zwischen $I + J$, $I \cdot J$ und I^2 .
- b) Beschreiben Sie das Ideal $\langle x^e \rangle$, wobei $e \in \mathbb{N}$ fest sei, in $\mathbb{Q}[[x]]$.
- c) Betrachten Sie den kanonischen Homomorphismus

$$\phi_{\langle x^e \rangle} : \mathbb{Q}[[x]] \rightarrow \mathbb{Q}[[x]]/\langle x^e \rangle.$$

Beschreiben Sie die Elemente des homomorphen Bildes in $\mathbb{Q}[[x]]/\langle x^e \rangle$. Geben Sie auch eine praktische Darstellung dieser Elemente an.

- d) Geben Sie eine Darstellung der Elemente von $\mathbb{Q}[x]/\langle x^2 + 1 \rangle$ an. Zeigen Sie, dass dieser Quotientenring ein Körper ist.
- e) Beschreiben Sie $\mathbb{R}[x]/\langle x^2 + 1 \rangle$.
- f) Ist $\mathbb{Z}[x]/\langle x^2 + 1 \rangle$ ein Körper? Ein Integritätsbereich? Beschreiben Sie den Zusammenhang zwischen diesem Quotientenring und den ganzen Gaußschen Zahlen $G = \{a + b \cdot \sqrt{-1} \mid a, b \in \mathbb{Z}\}$.

zu **Aufgabe22:**

a) Anwendung von Aufgabe 5b)

```

> with(Domains);
> Qx:=DenseUnivariatePolynomial(Q,x);
> g:=Qx[Input](x^3-x^2+2);
      g := x^3 - x^2 + 2
> h:=Qx[Input](x^2+x+1);
      h := x^2 + x + 1
> Qx[Gcdex](g,h,'s','t');
      1
> s,t;
      3/13 - 1/13 x, 7/13 - 5/13 x + 1/13 x^2
> f:=Qx[Input](x^4+2*x);
      f := x^4 + 2 x
> Qx['+'](Qx['*'](f,s,g),Qx['*'](f,t,h));
      x^4 + 2 x
> sp:=Qx['*'](s,f);
      sp := 6/13 x - 2/13 x^2 + 3/13 x^4 - 1/13 x^5
> tp:=Qx['*'](t,f);
      tp := 14/13 x - 10/13 x^2 + 2/13 x^3 + 7/13 x^4 - 5/13 x^5 + 1/13 x^6
> Qx['+'](Qx['*'](sp,g),Qx['*'](tp,h));
      x^4 + 2 x
> sigma:=Qx[Rem](sp,h);%$ '%i$
      sigma := 3/13 + 12/13 x
> q:=Qx[Quo](sp,h);
      q := -3/13 - 3/13 x + 4/13 x^2 - 1/13 x^3
> tau:=Qx['+'](tp,Qx['*'](q,g));
      tau := -6/13 + 8/13 x + 1/13 x^2
> Qx['+'](Qx['*'](sigma,g),Qx['*'](tau,h));
      x^4 + 2 x

```

b)

```

> Qx:=DenseUnivariatePolynomial(Q,x);
> m1:=Qx[Input](x^3+x^2-1);
      m1 := x^3 + x^2 - 1
> m2:=Qx[Input](x^2+2*x+2);
      m2 := x^2 + 2 x + 2

```

```

> Qx[Gcdex](m2,m1,'b1','b11');
1
> l1:=Qx['*'](b1,m2);
l1 := 2 - x^2 - x^3
> l2:=Qx['*'](b11,m1);
l2 := x^3 + x^2 - 1
> Qx[Rem](l1,m1),Qx[Rem](l1,m2),Qx[Rem](l2,m1),Qx[Rem](l2,m2);
1, 0, 0, 1
> r1:=Qx[Input](2*x^2+1); r2:=Qx[Input](x+2);
r1 := 2x^2 + 1
r2 := x + 2
> r:=Qx['+'](Qx['*'](r1,l1),Qx['*'](r2,l2));
r := -x + 5x^2 + 2x^3 - x^4 - 2x^5
> Qx[Rem](r,m1),Qx[Rem](r,m2);
2x^2 + 1, x + 2

> Qx[Rem](r,Qx['*'](m1,m2));
-4 - 5x + 7x^2 + 10x^3 + 5x^4

```

zu **Aufgabe23:**

a) Mit den Abkürzungen $c_{ij} = m_i^{-1} \bmod m_j$ und $C_j = \prod_{1 \leq i < j} c_{ij} \bmod m_j$ schreibt sich die Variante aus der Vorlesung als $\nu_k \equiv \left(u_k - \left(\nu_0 + \nu_1 m_0 + \dots + \nu_{k-1} \prod_{i=0}^{k-2} m_i \right) \right) C_k \pmod{m_k}$. Durch Ausmultiplizieren und neues Zusammenfassen erhält man die gesuchte Darstellung.

b) Es werden im Schritt für ν_k k Inverse benötigt, also insgesamt $n(n-1)/2$ Inverse c_{ij} mit $i < j$.

c) Beide kann man analog dem Horner Schema auswerten, so dass für ν_k jeweils k Additionen und k Multiplikationen erforderlich sind. Um in der Darstellung aus der Vorlesung die wiederholte Berechnung von $m_i \bmod m_j$ zu vermeiden, könnte man eine Tabelle dieser m_{ij} anlegen oder die Einschränkung $m_0 < m_1 < \dots < m_k < \dots$ treffen.

Also bleibt als entscheidender Unterschied die Berechnung der Inversen im ersten Schritt. Sei $s = \log \max m_j$. Für die c_{ij} benötigen wir $n(n-1)/2$ Inversenbildungen modulo ein m_j in Form von Berechnungen des ggT. Das sind $O(n^2 M(s) \log(s))$ Wortoperationen. Für die C_j werden $n(n-1)/2$ Multiplikationen und n Inversenbildungen modulo ein m_j benötigt. Zusammen sind das $O(n^2 M(s) + n M(s) \log(s))$ Wortoperationen. Also ist das Verfahren aus der Vorlesung vorzuziehen.

zu **Aufgabe24:**

```

> with (Domains);
> Zx:=DenseUnivariatePolynomial(Z,x);
> <br>
> Z7x:=DenseUnivariatePolynomial(Zmod(7),x);

```

```

> p:=Zx[Input](x^6-531*x^5+94137*x^4-5598333*x^3+4706850*x^2-1327500*x+125000);
  p := x^6 - 531 x^5 + 94137 x^4 - 5598333 x^3 + 4706850 x^2 - 1327500 x + 125000
> p7:=Z7x[Input](x^6-531*x^5+94137*x^4-5598333*x^3+4706850*x^2-1327500*x+125000);
      p7 := 1 + x + x^2 + x^3 + x^4 + x^5 + x^6
> p7w:=Z7x[Input](x^2-2*x+1);
      p7w := 1 + 5 x + x^2
> Z7x['^'](p7w,3);
      1 + x + x^2 + x^3 + x^4 + x^5 + x^6
> u[1]:=p7w;
      u1 := 1 + 5 x + x^2
> p7du1:=Z7x['*'](-3,Z7x['^'](p7w,2));
      p7du1 := 4 + 5 x + 3 x^2 + 5 x^3 + 4 x^4
> for i from 1 to 5 do
> <br> Fu[i]:=Zx['-'](p,Zx['^'](u[i],3));
> <br> z:=Z7x[Input](Zx[Output](Zx[Div](Fu[i],7^i)));
> <br> u[i+1]:=DenseUnivariatePolynomial(Zmod(7^(i+1)),x)[Input](
> <br>   Zx[Output](Zx['-'](u[i],Zx['*'](
> <br>     7^(i),
> <br>     Zx[Input](
> <br>       Z7x[Output](
> <br>         Z7x[Div](
> <br>           z,
> <br>           p7du1))))));
> <br>end do;
> <br>
Fu1 := 124999 - 1327515 x + 4706772 x^2 - 5598488 x^3 + 94059 x^4 - 546 x^5
z := 6 x + 4 x^2 + x^3 + 4 x^4 + 6 x^5
u2 := 1 + 19 x + x^2
Fu2 := 124999 - 1327557 x + 4705764 x^2 - 5605306 x^3 + 93051 x^4 - 588 x^5
z := 3 + 4 x + 3 x^2 + 2 x^4 + 2 x^5
u3 := 50 + 166 x + x^2
Fu3 := -2572500 x + 565950 x^2 - 10222429 x^3 + 11319 x^4 - 1029 x^5
z := 4 x + 5 x^2 + 3 x^3 + 5 x^4 + 4 x^5
u4 := 50 + 2224 x + x^2
Fu4 := -18007500 x - 737227050 x^2 - 11006560957 x^3 - 14744541 x^4 - 7203 x^5
z := 4 x + 5 x^2 + 3 x^3 + 5 x^4 + 4 x^5
      u5 := 50 + 16630 x + x^2
> test:=Zx[Input](x^2-177*x+50);
      test := 50 + x^2 - 177 x
> Zx['^'](test,3);
      x^6 - 531 x^5 + 94137 x^4 - 5598333 x^3 + 4706850 x^2 - 1327500 x + 125000

```

Auch wenn es nicht so scheint, ist die richtige Lösung berechnet worden. Warum?
 Aber es geht dann doch noch anders, auch wenn es nicht so ganz einfach ist, Maple davon zu überzeugen

```
> p:=x^6-531*x^5+94137*x^4-5598333*x^3+4706850*x^2-1327500*x+125000;
  p := x6 - 531 x5 + 94137 x4 - 5598333 x3 + 4706850 x2 - 1327500 x + 125000
> p7w:=mods(x^2-2*x+1,7)
```

Warning, inserted missing semicolon at end of statement, ... (x²-2*x+1,7);

$$p7w := x^2 - 2x + 1$$

```
> u[1]:=p7w;
```

$$u_1 := x^2 - 2x + 1$$

```
> p7du1:=mods(-3*(p7w^2),7);
```

$$p7du1 := -3 (x^2 - 2x + 1)^2$$

```
> for i from 1 to 5 do
> <br> Fu[i]:=mods(evala(p-u[i]^3),7^(i+1));
> <br> z:=mods((Fu[i]/7^i),7);
> <br> u[i+1]:=u[i] - mods(Quo(z,p7du1,x),7)*7^i;
> <br>end do;
> <br>
```

$$Fu_1 := 14x^5 - 7x^4 - 14x^3 - 7x^2 + 14x$$

$$z := 2x^5 - x^4 - 2x^3 - x^2 + 2x$$

$$u_2 := x^2 + 19x + 1$$

$$Fu_2 := 98x^5 + 98x^4 + 147x^2 - 147x + 147$$

$$z := 2x^5 + 2x^4 + 3x^2 - 3x + 3$$

$$u_3 := x^2 + 166x + 50$$

$$Fu_3 := -1029x^5 - 686x^4 + 1029x^3 - 686x^2 - 1029x$$

$$z := -3x^5 - 2x^4 + 3x^3 - 2x^2 - 3x$$

$$u_4 := x^2 - 177x + 50$$

$$Fu_4 := 0$$

```
> simplify(p-u[6]^3);
```

0

zu Aufgabe25:

- Man erhält $I + J = \langle x, y \rangle$, $I \cdot J = \langle x^2, xy \rangle$ und $I^2 = \langle x^2, xy, y^2 \rangle$. Anhand der Inklusionseigenschaften der Erzeugenden sieht man, dass $J \subset I$, $I \cdot J \subset I^2 \subset I = I + j$.
- Das Ideal enthält alle Potenzreihen der Ordnung e oder größer.
- Jede Potenzreihe lässt sich in eine Summe aus einem Polynom höchstens vom Grad $e - 1$ und einem Element des Ideals zerlegen. Die Elemente des homomorphen Bildes sind damit Polynome vom Grad höchstens $e - 1$.
- $a + bx$ mit $a, b \in \mathbb{Q}$, Division mit Rest. Irreduzibilität von $x^2 + 1$ in \mathbb{Q} , dann Satz.
- Isomorph zu \mathbb{C} , ähnlich d)

f) Ist kein Körper, nur wenige Elemente sind Einheiten. Ist Integritätsbereich, kann man durch Betrachten der Betragsfunktion zeigen, $|a + bx|^2 = a^2 + b^2$. Der Betrag ist genau dann null, wenn a und b null sind. Der Betrag des Produkts ist das Produkt der Beträge, d.h. die Betragsfunktion ist ein Homomorphismus auf eine Teilmenge der reellen Zahlen mit Kern $\{0\}$. Ein Produkt kann also nur dann 0 sein, wenn es einer der Faktoren schon ist.

Übungen zur Vorlesung Computeralgebra
Blatt 7

Prof. Dr. Klaus Madlener

26. Aufgabe:

Betrachten Sie den Algorithmus aus der Vorlesung (F 102) zur Bestimmung eines polynomialen Inversen $g \in D[x]$ modulo x^ℓ ($\ell \in \mathbb{N}$) zu gegebenem $f \in D[x]$ mit $f(0) = 1$. Zeigen Sie, dass, wenn $\ell = 2^r$ eine Zweierpotenz ist, die Rechenzeit des Algorithmus höchstens $3M(\ell) + \ell \in O(M(\ell))$ Operationen in D beträgt. ($M(n)$ bezeichne die Rechenzeit einer Multiplikation zweier Polynome vom Grade $\leq n$.)

Wie könnte man vorgehen, um im Falle, dass ℓ keine Zweierpotenz ist, die Berechnung zu vieler Koeffizienten des polynomialen Inversen zu vermeiden?

Zeigen Sie weiter, dass die Rechenzeit des Algorithmus aus der Vorlesung auf $2M(\ell)$ Operationen in D fällt wenn $\text{char}(D) = 2$ ist.

27. Aufgabe:

Sei R ein kommutativer Ring mit 1, $F \in R[y]$, $g \in R$, wobei $F(g) \equiv 0 \pmod{p}$ und $F'(g)$ invertierbar modulo p sei, sei eine Anfangslösung, und $\ell \in \mathbb{N}^+$.

Zeigen Sie: Wenn $h, h^* \in R$ Lösungen modulo p^ℓ mit $h \equiv g \equiv h^* \pmod{p}$ sind und $F(h) \equiv 0 \equiv F(h^*) \pmod{p^\ell}$ gilt, so ist $h \equiv h^* \pmod{p^\ell}$.

28. Aufgabe:

Berechnen Sie in $\mathbb{Z}_3[x, y, z]$ mit Hilfe der ideal-adischen Iteration und dem Ideal

$$I = \langle y - 1, z \rangle \subseteq \mathbb{Z}_3[x, y, z]$$

die Lösung der Gleichung

$$u^2 - u = x^6 + x^4y^2 + 2x^3z + x^2y^4 + xy^2z + z^2 + 2.$$

Bestimmen Sie dazu zunächst die I -adische Darstellung des Polynoms auf der rechten Seite.

zu **Aufgabe26:**

Sei $r = \lceil \log l \rceil$. Der Algorithmus führt r Schritte aus. In jedem Schritt wird ein g_i berechnet, wozu $M(2^{i-1})$ Koeffizientenoperationen für g_{i-1}^2 und $M(2^i)$ für das Produkt fg_{i-1}^2 . Die obere Hälfte von g_i ist gleich der negierten oberen Hälfte dieses Produkts, also sind noch 2^{i-1} Operationen dafür zu berechnen. Für Schritt i sind das dann $M(2^i) + M(2^{i-1}) + 2^{i-1} \leq \frac{3}{2}M(2^i) + 2^{i-1}w$ Koeffizientenoperationen. Zusammen ergibt sich:

$$\sum_{1 \leq i \leq r} \left(\frac{3}{2}M(2^i) + 2^{i-1} \right) \leq \left(\frac{3}{2}M(2^r) + 2^{r-1} \right) \sum_{1 \leq i \leq r} 2^{i-r} < 3M(2^r) + 2^r = 3M(l) + l$$

Falls l keine Zweierpotenz ist, könnte man in Schritt i modulo $\lceil l/2^{r-i+1} \rceil$ rechnen. Dabei würde man in jedem Schritt höchstens einen Koeffizienten „verschenken“. Z.B. bei $l = 9$ würde man nacheinander $(\text{mod } x)$, $(\text{mod } x^2)$, $(\text{mod } x^3)$, $(\text{mod } x^5)$, $(\text{mod } x^9)$ rechnen. Wenn man das analysiert, stellt man fest, dass dies keinen großen Nachteil für die Laufzeit bringt. Für Schritt $i = r - j$ erhalten wir $M(\lceil l/2^j \rceil) + M(\lceil l/2^{j-1} \rceil) + \lceil l/2^{j-1} \rceil \leq \frac{3}{2}M(\lceil l/2^j \rceil) + \lceil l/2^{j-1} \rceil w$ Koeffizientenoperationen. Damit gilt dann, wenn man nur die Multiplikationen betrachtet, weil die Subtraktionen dagegen nicht ins Gewicht fallen:

$$\frac{3}{2} \sum_{0 \leq j < r} M(\lceil l/2^j \rceil) < \frac{3}{2}M(l) \sum_{0 \leq j < r} 2^{-j} < 3M(l)$$

Falls $\text{char}(D) = 2$ vereinfacht sich die Rekursionsformel zu $g_i \equiv fg_{i-1}^2$, weil $2 = 1 + 1 = 0$ nach Voraussetzung.

zu **Aufgabe27:**

Es ist zu zeigen, dass die p -adische Iteration in jedem Schritt eindeutige Approximationen modulo p^l liefert. Induktion über l . Induktionsanfang nach Voraussetzung. Induktionsschritt $l - 1 \rightarrow l$. Seien h und h^* Lösungen modulo p^l , dann wurden sie durch einen Schritt aus den Lösungen k und k^* modulo p^{l-1} geliftet. Für k und k^* gilt die Induktionsvoraussetzung, also gilt $k \equiv k^* \pmod{p^{l-1}}$. Da alle Voraussetzungen erfüllt sind, können wir einen linearen Newton-Schritt anwenden, und erhalten so h und h^* und es gilt $F(h) \equiv F(h^*) \equiv 0 \pmod{p^l}$. Damit gilt $F(h) - F(h^*) = F(k) - F(k^*) + F'(k)u(x)p^{l-1} - F'(k^*)u^*(x)p^{l-1} + wp^l \equiv 0 \pmod{p^l}$. Dann muss aber schon $F'(k)u(x) - F'(k^*)u^*(x) \equiv 0 \pmod{p}$ gelten. Da $F'(k) = F'(k^*) \pmod{p^{l-1}}$ muss $u(x) \equiv u^*(x) \pmod{p}$ gelten. Damit gilt $h \equiv h^* \pmod{p^l}$.

zu **Aufgabe28:**

Wir haben das folgende $F(u)$, um über $\mathbb{Z}_3[x, y, z]$ eine Lösung mit $F(u) = 0$ zu bestimmen und wir haben $I = \langle y - 1, z \rangle$ gegeben.

$$\begin{aligned} F(u) &= u^2 - u - (x^6 + x^4y^2 + 2x^3z + x^2y^4 + xy^2z + z^2 + z) \\ &= u^2 - u - (x^2(y - 1)^4 + x^2(y - 1)^3 + x^4(y - 1)^2 + xz(y - 1)^2 + x^2(y - 1) + 2x^4(y - 1) \\ &\quad + 2xz(y - 1) + z^2 + 2x^3z + xz + x^6 + x^4 + x^2 + 2) \end{aligned}$$

Dabei haben wir $F(u)$ in die I -adische Darstellung gebracht. Für die Startlösung $\text{mod } I^2$ erhält man durch geschicktes Ausprobieren (max. 80 Möglichkeiten) $u^{(1)} = x^3 + 2x + 2$.

Wir haben $F'(u) = 2u - 1$, $F'(u^{(1)}) = 2x^3 + x$. Damit sind alle Voraussetzungen zum Lifting erfüllt.

$$\begin{aligned}
F(u^{(1)}) &= (x^3 + 2x + 2)^2 - x^3 - 2x - 2 - \dots \bmod I^2 \\
&\equiv x^6 + 2x^4 + 2x^3 + 2x^4 + 4x^2 + 4x + 2x^3 + 4x + 4 - x^3 - 2x - 2 \\
&\quad - x^2(y-1) - 2x^4(y-1) - 2x^3z - xz - x^6 - x^4 - x^2 - 2 \\
&= -x(2x^3 + x)(y-1) - (2x^3 + x)z \\
u^{(2)} &= u^{(1)} + x(y-1) + z = x^3 + 2x + 2 + x(y-1) + z \\
F(u^{(2)}) &= ((x^3 + 2x + 2) + (x(y-1) + z))^2 - x^3 - 2x - 2 - x(y-1) - z - (\dots) \bmod I^3 \\
&\equiv 2x^4(y-1) + 2x^3z + 4x^2(y-1) + 4xz + 4x(y-1) + 4z + x^2(y-1)^2 + 2x(y-1)z + z^2 \\
&\quad - x(y-1) - z - x^4(y-1) - x^2(y-1) - 2x^4(y-1) - 2x(y-1)z - z^2 - 2x^3z - xz \\
&= x(2x^3 + x)(y-1)^2 \\
u^{(3)} &= u^{(2)} - x(y-1)^2 = x^3 + 2x + 2 + x(y-1) + z - x(y-1)^2 \\
F(u^{(3)}) &= (u^{(2)} - x(y-1)^2)^2 - u^{(2)} + x(y-1)^2 - (\dots) \bmod I^4 \\
&\equiv -2u^{(2)}x(y-1)^2 + \overline{x^2(y-1)^4} + x(y-1)^2 - (\dots) \\
&= -2x^4(y-1)^2 - 4x^2(y-1)^2 - 4x(y-1)^2 - 2x^2(y-1)^3 - 2x(y-1)^2z + x(y-1)^2 \\
&\quad + 2x^4(y-1)^2 + x^2(y-1)^2 - x^2(y-1)^3 - x(y-1)^2z \\
&= 0 \\
u^{(4)} &= u^{(3)} = x^3 + 2x + 2 + x(y-1) + z - x(y-1)^2 \\
F(u^{(4)}) &= F(u^{(3)}) \bmod I^5 \\
&= x^2(y-1)^4 - x^2(y-1)^4 = 0
\end{aligned}$$

Bei der Berechnung der $F(u^{(i)})$ wurden die Terme weggelassen, die sich bei der jeweils vorangehenden Berechnung schon gegenseitig ausgelöscht haben. Bei $F(u^{(4)})$ haben sich alle Terme ausgelöscht.

Übungen zur Vorlesung Computeralgebra
Blatt 8

Prof. Dr. Klaus Madlener

29. Aufgabe:

Zeigen Sie den Satz 5.18 auf Folie 260: Der Algorithmus für den quadratischen Hensel-Lifting-Schritt ist korrekt.

30. Aufgabe:

Betrachten Sie das Faktorisierungsbeispiel aus der Vorlesung: Sei $a(x) = 12x^3 + 10x^2 - 36x + 35 \in \mathbb{Z}[x]$; eine Faktorisierung modulo 5 ist

$$\phi_5(a(x)) = 2 \cdot x \cdot (x^2 + 2) \in \mathbb{Z}_5[x].$$

Wenden Sie quadratisches Hensel-Lifting (s. F 257f.) an, um eine Faktorisierung von $a(x)$ in $\mathbb{Z}[x]$ zu berechnen.

31. Aufgabe:

Wir definieren für ein Polynom $f = \sum_{0 \leq i \leq n} f_i x^i = f_n \prod_{1 \leq i \leq n} (x - z_i) \in \mathbb{C}[x]$ das Landau-Maß $M(f)$ durch

$$M(f) = |f_n| \cdot \prod_{1 \leq i \leq n} \max\{1, |z_i|\},$$

wobei $f_0, \dots, f_n, z_1, \dots, z_n \in \mathbb{C}$. Weiter definieren wir noch die Maximumsnorm $\|f\|_\infty$, die 1-Norm $\|f\|_1$ und die 2-Norm $\|f\|_2$ durch

$$\begin{aligned} \|f\|_\infty &= \max_{0 \leq i \leq n} |f_i| \\ \|f\|_1 &= \sum_{0 \leq i \leq n} |f_i| \\ \|f\|_2 &= \sqrt{\sum_{0 \leq i \leq n} |f_i|^2}. \end{aligned}$$

Dabei ist $|a| = \sqrt{a\bar{a}}$ für $a \in \mathbb{C}$ (\bar{a} ist die \mathbb{C} -Konjugierte zu a).

- a) Zeigen Sie, dass für jedes $f \in \mathbb{C}[x]$ gilt: (a.1) $M(f) \geq |f_n|$, (a.2) $M(f) = M(g)M(h)$, falls $f = gh$ mit $g, h \in \mathbb{C}[x]$, und (a.3) $M(f) \leq \|f\|_2$.
- b) Wenn $h = \sum_{0 \leq i \leq m} h_i x^i \in \mathbb{C}[x]$ vom Grad m ein Teiler von $f = \sum_{0 \leq i \leq n} f_i x^i \in \mathbb{C}[x]$ vom Grad $n \geq m$ ist, so gilt:

$$\|h\|_2 \leq \|h\|_1 \leq 2^m M(h) \leq 2^m \left| \frac{h_m}{f_n} \right| \|f\|_2.$$

- c) Nun zum eigentlichen Ziel: Seien $f, g, h \in \mathbb{Z}[x]$ mit $\deg f = n \geq 1$, $\deg g = m$, $\deg h = k$ und es sei gh ein Teiler von f in $\mathbb{Z}[x]$. Zeigen Sie

$$\|g\|_\infty \|h\|_\infty \leq 2^{m+k} \|f\|_2 \leq \sqrt{n+1} \cdot 2^{m+k} \|f\|_\infty$$

sowie

$$\|h\|_\infty \leq \sqrt{n+1} \cdot 2^k \|f\|_\infty.$$

32. Aufgabe:

Sei $f(x, y) = f_0x^d + f_1x^{d-1}y + f_2x^{d-2}y^2 + \dots + f_dy^d$ ein bivariates homogenes Polynom. Geben Sie eine Reduktion des Faktorisierungsproblems für solche Polynome auf das Faktorisierungsproblem für univariate Polynome an.

zu **Aufgabe29:**

Zunächst zu den Behauptungen über u^*, w^* : Wegen $1 - su - tw \equiv e \equiv 0 \pmod{m}$ und $q \equiv 0 \pmod{m}$ gilt:

$$\begin{aligned} a - u^*w^* &\equiv a - (u + te + qu)(w + se - qw) \\ &= a - uw - (su + tw)e - ste^2 - (su + tw)qe + uwq^2 \\ &\equiv (1 - su - tw)e - ste^2 - (su - tw)qe + uwq^2 \equiv 0 \pmod{m^2} \end{aligned}$$

Nach Konstruktion ist $\deg(r) < \deg(w)$ und w monisch, also ist auch w^* monisch und vom selben Grad wie w . Da $a \equiv u^*w^* \pmod{m^2}$ und w^* monisch, gilt $\deg(u^*) = \deg(a) - \deg(w^*)$ und $\deg(u) = \deg(a) - \deg(w)$.

Für die Behauptungen mit s^*, t^* hat man

$$\begin{aligned} s^*u^* + t^*w^* - 1 &= (s - sb + cw^*)u^* + (t - tb - cu^*)w^* - 1 \\ &= su^* + tw^* - 1 - (su^* + tw^*)b \\ &= b - (b - 1) * b \equiv -b^2 \equiv 0 \pmod{m^2}, \end{aligned}$$

wegen $b \equiv 0 \pmod{m}$. Mit dem Altlemma (Folie 258) folgt, dass $c, d \equiv 0 \pmod{m}$. Daraus folgt $s^* \equiv s \pmod{m}$ und $t^* \equiv t \pmod{m}$

Wegen $\deg(d) < \deg(w^*) = \deg(w)$ und $\deg(s) < \deg(w)$ gilt $\deg(s^*) < \deg(w^*)$. Wegen w^* monisch und $s^*u^* + t^*w^* \equiv 1 \pmod{m^2}$ gilt $\deg(t^*) = \deg(s^*) + \deg(u^*) - \deg(w^*) < \deg(u^*)$.

zu **Aufgabe30:**

$$a(x) = 12x^3 + 10x^2 - 36x + 35$$

$$\Phi_5(a(x)) = (2x)(x^2 + 2) = uw$$

$$x(2x) + 3(x^2 + 2) \equiv 1 \pmod{5} \quad s = x, t = 3$$

$$e^{(1)} = a - uw \equiv 10x^3 + 10x^2 - 15x + 10 \pmod{25}$$

$$se \equiv 10x^4 + 10x^3 - 15x^2 + 10x = (x^2 + 2)(10x^2 + 10x - 10) + 5x + 20$$

$$u^* = u + te + qu \equiv 12x + 5 \pmod{25} \quad w^* = w + r \equiv x^2 - 10x - 3 \pmod{25}$$

$$b = su + tw - 1 = x(12x + 5) + 3(x^2 - 10x - 3) - 1 \equiv 15x^2 - 10 \pmod{25}$$

$$sb = 15x^3 - 10x = (x^2 - 10x - 3)(15x) + 35x = cw^* + d$$

$$s^* = s - d \equiv -9x \pmod{25}$$

$$t^* = t - te + qu^* = 3 - 45x^2 + 30 - 15x(12x + 5) \equiv 8 \pmod{25}$$

$$s^*u^* + t^*w^* = -100x^2 - 125x - 24 \equiv 1 \pmod{25} \quad \text{Probe!}$$

$$e^{(2)} = a - uw \equiv 125x^2 + 50x + 50 \pmod{625}$$

$$se \equiv 125x^3 + 175x^2 + 175x \pmod{625}$$

$$\equiv (x^2 - 10x - 3)(125x + 175) - 200x - 100 \pmod{625}$$

$$u^* = u + te + qu \equiv 12x + 30 \pmod{625} \quad w^* = w + r \equiv x^2 - 210x - 103 \pmod{625}$$

An dieser Stelle wissen wir, dass u^*, w^* assoziiert zur Lösung mod 625 sein muss. Wir erhalten $\bar{u} = u^*6^{-1} \equiv 2x + 5 \pmod{625}$ und $\bar{w} = 6w^* = 6x^2 - 10x + 7 \pmod{625}$. Dieses Aufteilen des Leitkoeffizienten könnte man auch algorithmisch machen, aber dazu müsste erstmal der Leitkoeffizient zerlegt werden. Besser ist, die modifizierte Hensel-Konstruktion für nicht monische Polynome anzuwenden. Also angenommen, wir wollten $12a(x)$ faktorisieren, dann können wir auch für w^* den Leitkoeffizienten 12 erhalten. Beim Bilden der primitiven Anteile muss diese 12 wieder aus beiden Faktoren verschwinden, da das $a(x)$ oBdA primitiv war. In der Tat ist $pp(12x + 30) = 2x + 5$ und $pp(12 * w^*) = pp(12x^2 - 20x + 14) = 6x^2 - 10x + 7$.

zu Aufgabe31:

Diese Aufgabe wird in Kapitel 6.6, Computer Algebra, von zur Gathen/Gerhard, gelöst. Das Ziel ist die Mignotte-Schranke (c). Mit dieser Schranke hat man Informationen darüber, wie weit man Liften muss, bzw. wie viele Homomorphismen man braucht, um das Ergebnis aus dem modularen Ergebnis bestimmen zu können.

Für Abschätzungen ist es im allgemeinen schöner, wenn man mit reellen oder komplexen Zahlen arbeiten kann. Bei Polynomen über den komplexen Zahlen hat man noch die schöne Eigenschaft, dass sie vollständig in Linearfaktoren zerfallen. Das alles wollen wir hier ausnutzen. Wir gewinnen zunächst Abschätzungen über die 2-Norm von Faktoren und Produkt (b) und daraus dann Abschätzungen über die Maximumsnorm (c) und damit über die maximale Größe der Koeffizienten. Für die Normen gilt dabei folgende Ungleichung $\|f\|_\infty \leq \|f\|_2 \leq \|f\|_1 \leq \sqrt{n+1}\|f\|_\infty$.

(a.1) und (a.2) folgen sofort aus der Definition des Landau-Maßes.

Abschätzung (a.3), die Landausche Ungleichung, geht elegant mit folgendem Lemma: Lemma (6.30) Für $f \in \mathbb{C}[x]$ und $z \in \mathbb{C}$ gilt: $\|(x - z)f\|_2 = \|(\bar{z}x - 1)f\|_2$.

Bei (b) ist es interessant zu betrachten, wie genau diese Abschätzung ist.

Man schreibt h in der Darstellung $h = h_m \prod_{1 \leq i \leq m} (x_i - u_i)$, wobei die $u_i \in \mathbb{C}$ auch Wurzeln von f sind.

Nach dem Satz von Vieta kann man die Koeffizienten von h in der gewöhnlichen Darstellung ausdrücken durch:

$$h_i = (-1)^{m-i} h_m \sum_{\substack{S \subseteq \{1, \dots, m\} \\ \#S = m-i}} \prod_{j \in S} u_j$$

Das führt dann zur folgenden Abschätzung für die Koeffizienten:

$$|h_i| \leq |h_m| \cdot \sum_S \prod_{j \in S} |u_j| \leq \binom{m}{i} M(h)$$

Diese Abschätzung ist in vielen Fällen sehr pessimistisch, deshalb wird die Schranke, die wir hier letztenendes bestimmen, in der Praxis zu Problemen führen, wenn man nicht vorsichtig mit ihr umgeht. Im letzten Schritt für (b) haben wir dann noch:

$$\|h\|_2 \leq \|h\|_1 = \sum_{0 \leq i \leq m} |h_i| \leq 2^m M(h) \leq \left| \frac{h_m}{f_n} \right| 2^m M(f) \leq \left| \frac{h_m}{f_n} \right| 2^m \|f\|_2$$

(c) folgt unmittelbar aus (b).

zu **Aufgabe32:**

Sei $f(x, y) = f_0x^d + f_1x^{d-1}y + f_2x^{d-2}y^2 + \dots + f_dy^d$. Falls $f(x, y) = g(x, y)h(x, y)$ für geeignete g, h , dann gilt auch $f(x, 1) = g(x, 1)h(x, 1)$. Umgekehrt, wenn $f(x, 1) = g(x)h(x)$, dann ist $y^d f(x/y, 1) = y^d g(x/y)h(x/y) = g(x, y)h(x, y) = f(x, y)$. Also lassen sich $g(x, y)$ und $h(x, y)$ aus einer Faktorisierung von $f(x, 1)$ in $g(x)$ und $h(x)$ bestimmen.

Übungen zur Vorlesung Computeralgebra
Blatt 9

Prof. Dr. Klaus Madlener

33. Aufgabe:

Zeigen oder widerlegen Sie:

- Das Polynom $x^{1000} + 2 \in \mathbb{Z}_5[x]$ ist quadratfrei.
- Sei F ein Körper und seien $f, g \in F[x]$. Dann ist der quadratfreie Anteil von fg das Produkt der quadratfreien Anteile von f und g .

Der quadratfreie Anteil eines Polynoms $h = \prod_{1 \leq i \leq r} h_i^{e_i}$ ist dabei $\prod_{1 \leq i \leq r} h_i$ (mit paarweise verschiedenen irreduziblen h_i).

34. Aufgabe:

Faktorisieren Sie das Polynom $x^8 + x^7 + 2x^6 + 3x^5 + 3x^4 + 3x^3 + 2x^2 + 2x + 1$ mit der Distinct-Degree-Methode jeweils über $\mathbb{Z}_7[x]$, $\mathbb{Z}_{19}[x]$ und $\mathbb{Z}_{23}[x]$.

35. Aufgabe:

Wie viele Faktoren hat $a(x) = x^4 + 1 \in \mathbb{Z}_p[x]$ (p prim), wenn (i) $p = 2$, (ii) $p \equiv 1 \pmod{8}$, (iii) $p \equiv 3 \pmod{8}$ bzw. (iv) $p \equiv 5 \pmod{8}$?

36. Aufgabe:

Machen Sie sich mit den Begriffen Sylvestermatrix und Resultante vertraut, z.B. Modern Computer Algebra, von zur Gathen, Kap. 6.3 S. 142-147, oder Algorithms for Computer Algebra, Geddes, Kap. 7.3, S. 285-288.

Zeigen Sie: Seien $f, g \in \mathbb{Z}[x]$, $r = \text{res}(f, g) \in \mathbb{Z}$, und $u \in \mathbb{Z}$. Dann gilt $\text{ggT}(f(u), g(u)) \mid r$.

37. Aufgabe:

Sei $p \in \mathbb{Z}$ prim und $n > 1$. Zeigen Sie die folgenden „pathologischen“ Eigenschaften von $R = \mathbb{Z}_{p^n}[x]$ ($f, g \in R$):

- Es gilt nicht notwendigerweise $\deg fg = \deg f + \deg g$.
- R ist kein ZPE-Ring.
- Es ist $\text{ggT}(f, g)$ nicht notwendigerweise als Linearkombination von f und g darstellbar.

zu **Aufgabe33:**

- a) $x^{1000} = (x^8 + 2)^{125} \pmod{5}$, also nicht quadratfrei. $f' = 0$, also ist f eine fünfte Potenz.
b) Ein Faktor kann in beiden quadratfreien Anteilen auftauchen, deren Produkt ist dann nicht mehr quadratfrei.

zu **Aufgabe34:**

Das Polynom zerfällt über \mathbb{Z}_7 und \mathbb{Z}_{19} in einen Faktor vom Grad 3, und 5 Faktoren vom Grad 1. über \mathbb{Z}_{23} zerfällt es in 2 vom Grad 1 und 3 vom Grad 2. Das demonstriert, dass es aussichtslos erscheint, mit einem Small-Primes-Ansatz und chinesischem Restsatz das Faktorisierungsproblem zu lösen. Die Anzahl der zu berücksichtigenden Faktorkombinationen die man bestimmen müsste, um den chinesischen Restsatz anwenden zu können, wird viel zu groß.

zu **Aufgabe35:**

(i) $p = 2$: Es ist $\text{ggT}(a, a') = a$, damit ist a nicht quadratfrei. Wir wissen $x^4 + 1 = x^4 + 1^4 = (x + 1)^4$ in \mathbb{Z}_2 .

(ii) $p \equiv 1 \pmod{8}$: $a' = 4x^3$. Es gilt $\text{ggT}(a, a') = 1$, wie man bei der Division mit Rest von a und a' im ersten Schritt erkennen kann. 4 ist sicher eine Einheit modulo p , wenn $4 \not\equiv 0 \pmod{p}$. Also ist a quadratfrei. Betrachtet man nun die Division mit Rest von $x^p - x$ und $x^4 + 1$, so erhält man $x^p - x = (x^4 + 1)(x^{p-4} - x^{p-8} \pm \dots) + x - x$, also Rest 0. Also hat a nur lineare Faktoren.

(iii) $p \equiv 3 \pmod{8}$: Quadratfreiheit von $x^4 + 1$ folgt wie bei (ii), wenn $p \neq 3$. Man berechnet die Restefolge $x^p - x, x^4 + 1, x^3 - x, x^2 + 1, -2x, 1$. Also ist $\text{ggT}(x^p - x, x^4 + 1) = 1$. Also gibt es keine linearen Faktoren. Für Faktoren vom Grad 2 berechnet man die Restefolge $x^{p^2} - x, x^4 + 1, 0$, weil $p^2 \equiv 1 \pmod{8}$ also muss es 2 Faktoren vom Grad 2 geben. Bei $p = 3$ liegt ein Quadrat eines Polynoms vom Grad 2 vor.

(iv) $p \equiv 5 \pmod{8}$: Quadratfreiheit von $x^4 + 1$ folgt wie bei (ii). Man berechnet die Restefolge $x^p - x, x^4 + 1, -2x, 1$, also keine linearen Faktoren. Für Faktoren vom Grad 2 ergibt sich wegen $p^2 \equiv 25 \equiv 1 \pmod{8}$ dieselbe Restefolge wie bei (iii), also gibt es auch hier 2 Faktoren vom Grad 2.

Für $p \equiv 7 \pmod{8}$ erhält man dasselbe Ergebnis wie bei (iii) und (iv) weil $p^2 \equiv 1 \pmod{8}$.

zu **Aufgabe36:**

$\mathbb{Z}[x]$ ist ein Hauptidealring. Das Ideal $\langle f, g \rangle$ wird von $\langle \text{ggT}(f, g) \rangle$ aufgespannt. Wenn $\text{res}(f, g) = 0$. bleibt nichts zu zeigen. Ansonsten ist $\text{ggT}(f, g) = 1$. Es existieren also eindeutige s, t mit den genannten Gradeigenschaften mit $sf + tg = 1$. Es gilt nun $\text{res}(f, g)s(u)f(u) + \text{res}(f, g)t(u)g(u) = \text{res}(f, g) \in \langle \text{ggT}(f(u), g(u)) \rangle$.

zu **Aufgabe37:**

(a) $(px)^n \equiv 0 \pmod{p^n}$

(b) $(x + p)^n = x(\sum_{0 \leq i < n} \binom{n}{i} p^{n-i} x^{i-1})$, wobei wir für die Summe eine beliebige Zerlegung annehmen dürfen. Der Faktor x rechts taucht links nicht auf und ist auch nicht zu $x + p$ oder einer Potenz davon assoziiert.

(c) $\text{ggT}(x + p, x - p) = 1$, weil beide irreduzibel sind und nicht um eine Einheit unterschieden sind. Betrachtet man das Ideal $\langle x - p, x + p \rangle$, dann ist $2p$ darin enthalten, aber sicher nicht 1. Wenn $s(x + p) + t(x - p) \equiv 1 \pmod{p^n}$, dann wäre $s + t \equiv 0 \pmod{p^n}$

und $p(s - t) \equiv 1 \pmod{p^n}$, letzteres ist nicht erfüllbar.

Übungen zur Vorlesung Computeralgebra
Blatt 10

Prof. Dr. Klaus Madlener

38. Aufgabe:

Sei $p \in \mathbb{N}$ eine Primzahl und $q = p^k$ für ein positives $k \in \mathbb{N}$, $f \in \mathbb{F}_q[x]$ ein monisches und quadratfreies Polynom vom Grade n sowie $R = \mathbb{F}_q[x]/\langle f \rangle$. Wir können den Frobenius-Endomorphismus $\alpha \mapsto \alpha^q$ von R über \mathbb{F}_q im Berlekamp-Algorithmus von F 182 durch den absoluten Frobenius-Endomorphismus $\alpha \mapsto \alpha^p$ von R über dem Primkörper \mathbb{F}_p ersetzen. Untersuchen Sie diese Variante und vergleichen Sie ihre Laufzeit mit der des ursprünglichen Algorithmus.

39. Aufgabe:

für $n \in \mathbb{N}^+$ sei

$$\Phi_n = \prod_{\substack{1 \leq k \leq n \\ \text{ggT}(k,n)=1}} (x - e^{2\pi i k/n}) = \prod_{\substack{\omega \in \mathbb{C} \text{ ist eine } n\text{-te} \\ \text{primitive EW}}} (x - \omega) \in \mathbb{C}[x]$$

das n -te Kreisteilungspolynom (siehe z. B. Heinz Lüneburg, *Galoisfelder, Kreisteilungskörper und Schieberegisterfolgen*, Bibliographisches Institut, 1979). Es gilt $\deg \Phi_n = \varphi(n)$.

a) Zeigen Sie: $x^n - 1 = \prod_{d|n} \Phi_d$.

b) Die Möbiusfunktion $\mu : \mathbb{N}^+ \rightarrow \{-1, 0, 1\}$ ist erklärt durch

$$\mu(n) = \begin{cases} 1 & \text{falls } n = 1, \\ (-1)^k & \text{falls } n \text{ das Produkt von } k \text{ verschiedenen Primzahlen ist,} \\ 0 & \text{falls } n \text{ nicht quadratfrei ist.} \end{cases}$$

Es gilt folgende Inversionsformel: Sei R ein kommutativer Ring mit 1 und $f, g : \mathbb{N}^+ \rightarrow R$ seien zwei Funktionen mit

$$f(n) = \sum_{d|n} g(d) \text{ für } n \in \mathbb{N}^+.$$

Dann gilt:

$$g(n) = \sum_{d|n} \mu\left(\frac{n}{d}\right) f(d) = \sum_{d|n} \mu(d) f\left(\frac{n}{d}\right) \text{ für } n \in \mathbb{N}^+.$$

Geben Sie nun unter Beachtung von a) eine Formel für Φ_n an.

c) Seien $n, k \in \mathbb{N}^+$. Dann gilt:

- (i) $\Phi_n = x^{n-1} + x^{n-2} + \dots + x + 1$, falls n prim ist.
 - (ii) $\Phi_{2n} = \Phi_n(-x)$, falls $n > 3$ und n ungerade ist.
 - (iii) $\Phi_{kn}\Phi_n = \Phi_n(x^k)$, falls k prim ist und n nicht teilt.
 - (iv) $\Phi_{kn} = \Phi_n(x^k)$, falls jeder Primteiler von k auch n teilt.
- d) Geben Sie unter Verwendung der Ergebnisse aus c) einen Algorithmus an, der aus $n \in \mathbb{N}^+$ und den verschiedenen Primteilern p_1, \dots, p_r von n das Polynom Φ_n berechnet. Ihr Algorithmus soll eine Laufzeit von $O(M(n) \log n)$ Operationen in \mathbb{Z} haben. (Zusatzfrage: Wieso gilt $\Phi_n \in \mathbb{Z}[x]$?)

40. Aufgabe:

1. Zeigen Sie das Eisenstein-Kriterium: Wenn $f \in \mathbb{Z}[x]$ und $p \in \mathbb{N}$ eine Primzahl, so dass $p \nmid \text{lc}(f)$, p alle anderen Koeffizienten von f teilt, und $p^2 \nmid f(0)$, dann ist f irreduzibel in $\mathbb{Q}[x]$.
2. Folgern Sie, dass für beliebige $n \in \mathbb{N}$ das Polynom $x^n - p$ irreduzibel in $\mathbb{Q}[x]$ ist.

zu **Aufgabe38:**

Der absolute Frobenius-Endomorphismus setzt eine Darstellung von \mathbb{F}_q als Vektorraum über \mathbb{F}_p voraus. Wir brauchen also ein Minimalpolynom μ vom Grad k und die Wurzel ω . Damit kann man dann $\omega^{ip} \bmod \mu$ mit $1 \leq i \leq k-1$ berechnen. Man berechnet $x^{jp} \bmod f$ mit $0 \leq j \leq n-1$. Die x^{jp} sind Vektoren der Dimension kn über \mathbb{F}_p . Zusammen hat man die kn Produkte $\omega^{ip}x^{jp}$, die zusammen die Matrix A des absoluten Frobenius-Endomorphismus bilden.

Um eine Basis des Kerns der Abbildung $A - I$ zu bestimmen, benötigt man maximal $(nk)^\nu$ Operationen in $\mathbb{F}(p)$.

Anschließend hat man wieder höchstens 2 ggT-Berechnungen modulo f mit Aufwand $M(n) \log nM(k)$ Operationen in \mathbb{F}_p , für die Potenz $a^{(p-1)/2}$ werden $M(n) \log pM(k)$ Operationen in \mathbb{F}_p benötigt.

Zusammen erhält man $(nk)^\nu + M(nk) \log p$ Operationen in \mathbb{F}_p .

für den relativen Frobenius-Endomorphismus hatten wir eine Komplexität von $n^\nu + M(n) \log q$ Operationen in \mathbb{F}_q , d.h. $n^\nu M(k) + M(nk)k \log p$ Operationen in \mathbb{F}_p . Man sieht, dass der absolute Frobenius-Endomorphismus dieser Rechnung nach mehr Arbeit für den Nullraum zu verrichten hat, aber für große k weniger Arbeit im zweiten Teil des Algorithmus zu verrichten hat.

In Bach, Shallit: Algorithmic Number Theory, Ex. 7.15–7.17, wo die Idee zu dieser Aufgabe herkommt, wird mit Annahme von $M(n) = n^2$ und einem Kunstgriff bei der Berechnung des Nullraums erreicht, dass die Berechnung für große k oder p zu Gunsten des absoluten Algorithmus ausgeht. Dabei kommt unter anderem heraus, dass der Aufwand für die Nullraumberechnung gleich sein soll, obwohl die Matrix bei der absoluten Variante um den Faktor k größer ist. Vorsicht bei Musterlösungen!

zu **Aufgabe39:**

a) Jede Wurzel ω von $x^n - 1$ ist eine n -te EW. Nach dem Satz von Lagrange teilt die Ordnung d von ω n , damit ist ω primitive d -te EW, also ist $\Phi_d(\omega) = 0$. Umgekehrt ist auch jede primitive d -te EW eine n -te EW, wenn $d|n$. Also haben beide Polynome dieselben Nullstellen.

Da $\text{ggT}(x^n - 1, (x^n - 1)') = 1$, ist $x^n - 1$ QF. Per Definition sind die Φ_d quadratfrei, außerdem auch paarweise teilerfremd. Also ist auch die rechte Seite QF. Beide Polynome sind monisch, QF und haben dieselben Nullstellen, also sind sie gleich.

b) Man zeigt zuerst, dass μ multiplikativ ist für teilerfremde Faktoren n, m ist, d.h. $\mu(nm) = \mu(n)\mu(m)$. für nicht quadratfreie Faktoren ist das klar. für quadratfreie, teilerfremde Faktoren ist es auch klar.

Ein weiteres Lemma: $\sum_{d|n} \mu(d) = 0$ für $n > 1$.

Dafübere zerlegt man n in $n = mp^e$, wobei $p \nmid m$ und $e \geq 1$. Dann ist $\sum_{d|n} \mu(d) = \sum_{d|m} \sum_{0 \leq i \leq e} \mu(dp^i) = \sum_{d|m} \sum_{0 \leq i \leq e} \mu(d)\mu(p^i) = \sum_{d|m} \mu(d)\mu(1) + \mu(d)\mu(p) = \sum_{d|m} \mu(d) - \mu(d) = 0$

Damit kommen wir zum angegebenen Ziel, denn:

$$\begin{aligned} \sum_{d|n} \mu(d) f\left(\frac{n}{d}\right) &= \sum_{d|n} \mu(d) \sum_{e|(n/d)} g(e) \\ &= \sum_{de|n} \mu(d) g(e) = \sum_{e|n} g(e) \sum_{d|(n/e)} \mu(d) = g(n), \end{aligned}$$

denn nur für $e = n$ ist $\sum_{d|(n/e)} \mu(d) \neq 0$.

Um das ganze dann auf die zyklotomischen Polynome anwenden zu können, siehe (a), muss man noch einen Schritt weiter gehen. Wenn $f(n) = \prod_{d|n} g(d)$ für $n \in \mathbb{N}^+$, dann gilt $\log f(n) = \sum_{d|n} \log g(d)$, dann mit dem bisher gezeigten $\log g(n) = \sum_{d|n} \mu\left(\frac{n}{d}\right) \log f(d)$. Das ergibt dann $g(n) = \prod_{d|n} f\left(\frac{n}{d}\right)^{\mu(d)}$.

Diese Formel auf (a) angewendet ergibt dann $\Phi_n = \prod_{d|n} (x^{n/d} - 1)^{\mu(d)}$

c) (i) Wenn n prim, dann ist $\Phi_n = \prod_{d|n, d} (x^{n/d} - 1)^{\mu(d)} = (x^n - 1)/(x - 1) = x^{n-1} + \dots + x + 1$.

(ii) Unter mehrfacher Anwendung von b) ergibt sich:

$$\begin{aligned} \Phi_{2n}(x) &= \prod_{d|2n} (x^{2n/d} - 1)^{\mu(d)} = \prod_{d|n} (x^{2n/d} - 1)^{\mu(d)} (x^{2n/2d} - 1)^{\mu(2d)} \\ &= \prod_{d|n} \frac{(x^{2n/d} - 1)^{\mu(d)}}{(x^{n/d} - 1)^{\mu(d)}} = \prod_{d|n} (x^{n/d} + 1)^{\mu(d)} \\ &= \prod_{d|n} ((-1)((-x)^{n/d} + 1))^{\mu(d)} = \prod_{d|n} (((-x)^{n/d} + 1))^{\mu(d)} \cdot (-1)^{\sum_{d|n} \mu(d)} = \Phi_n(-x) \end{aligned}$$

Eine andere Möglichkeit ergibt sich durch Betrachtung der Wurzeln der Polynome. Ist ω eine primitive n -te EW, dann ist $-\omega$ eine $2n$ -te primitive EW. Also haben beide Polynome die gleiche Anzahl Wurzeln, nur durch ihre Vorzeichen unterschieden. Außerdem sind sie monisch und QF, also sind sie gleich.

(iii) Zu den Graden der Polynome: $\phi(kn) + \phi(n) = (k-1)\phi(n) + \phi(n) = k\phi(n)$.

Zu den Wurzeln: Sei ω eine primitive kn -te EW, dann ist ω^k eine n -te primitive EW. Sei ω eine n -te EW, dann ist ω^k eine n -te primitive EW, beide Male vorausgesetzt, dass k prim und kein Teiler von n . Also sind alle Wurzeln des linken Polynoms auch Wurzeln des rechten Polynoms, ihre Grade sind gleich, beide sind monisch, also sind sie gleich.

(iv) Falls jeder Primteiler von k auch n teilt, gilt $\phi(kn) = k\phi(n)$. Also sind die Grade der Polynome gleich. Wie bei (iii) erhält man, dass die Wurzeln der linken Seite auch Wurzeln der rechten Seite sind, also ist wieder alles gezeigt.

d) Wenn p_1, \dots, p_r die Primfaktoren von n sind. Dann können wir c)(iii) ausnutzen, um für $j = 1, \dots, r$ nacheinander $\Phi_{p_1 \dots p_j}$ zu berechnen, es gilt $\Phi_{p_1 \dots p_j p_{j+1}} = \Phi_{p_1 \dots p_j}(x^{p_{j+1}}) / \Phi_{p_1 \dots p_j}$. Am Ende kann durch Anwendung von c)(iv) $\Phi_n = \Phi_{p_1 \dots p_r(n/(p_1 \dots p_r))} = \Phi_{p_1 \dots p_r}(x^{n/(p_1 \dots p_r)})$ bestimmen.

Zur Komplexität: Es gibt maximal $\log_2(n)$ verschiedene Primfaktoren. für jeden dieser Faktoren haben wir eine Polynomdivision vom Grad höchstens n zu machen. Also haben wir $O(M(n) \log n)$ Operationen in \mathbb{Z} .

zu **Aufgabe40:**

1. Angenommen ein Polynom a erfhrt die genannten Bedingungen, lsst sich aber faktorisieren in $a_n x^n + \dots + a_0 = (b_m x^m + \dots + b_0)(c_{n-m} x^{n-m} + \dots + c_0)$, wobei $b, c \in \mathbb{Z}[x]$. Dann gilt $p|a_0 = b_0 c_0$, aber $p^2 \nmid b_0 c_0$. Dann teilt p oBdA c_0 , aber nicht b_0 . Wir wissen, dass $a_i = \sum_{0 \leq j \leq i} b_j c_{i-j}$. Wegen $p|a_i$ fr $i < n$, insbesondere fr $i \leq n - m$ folgt nun der Reihe nach, dass $p|c_i$. Damit gilt $p|c$ und $p|a$, Widerspruch zu $p \nmid a_n$.

Jede Faktorisierung ber \mathbb{Q} liee sich durch Erweitern so zu einem Polynom ber \mathbb{Z} umwandeln, dass die Voraussetzungen des Produkts noch erfhrt wren.

2. $x^n - p$ erfhrt die Voraussetzungen, ist also irreduzibel in $\mathbb{Q}[x]$.

41. Aufgabe:

Sei $f \in \mathbb{Z}[x]$ vom Grad n und die Maximumsnorm $\|f\|_\infty = A$ und $f = (ux + v)g$, wobei $u, v \in \mathbb{Z} \setminus \{0\}$ und $g = \sum_{0 \leq i < n} g_i x^i \in \mathbb{Z}[x]$.

1. Zeigen Sie, dass $|g_i| < (i + 1)A/|v|$ für $0 \leq i < n - 1$, falls $|u| = |v|$, und folgern Sie, dass dann $\|g\|_\infty \leq nA$.
2. Angenommen $\alpha = |u/v| < 1$. Zeigen Sie $|g_i| \leq A \frac{1 - \alpha^{i+1}}{1 - \alpha} / |v|$ für $0 \leq i < n - 1$, und folgern Sie, dass dann $\|g\|_\infty \leq A$ gilt. Zeigen Sie, dass letzteres auch im Fall $|u/v| > 1$ gilt.

42. Aufgabe:

Betrachten Sie die deterministische Variante des Berlekamp-Algorithmus in Geddes et al. auf Seite 352 (Algorithm 8.4). Wieso genügt es im letzten Abschnitt, die größten gemeinsamen Teiler $\text{ggT}(v^{[r]} - s, u)$ für die Basispolynome $v^{[2]}, \dots, v^{[k]}$ der Nullraumbasis von $Q - I$ zu berechnen, um eine vollständige Faktorisierung zu erhalten?

43. Aufgabe:

Von Kronecker (1882) stammt folgende Methode, das Faktorisierungsproblem für multivariate Polynome über einem ZPE-Ring R auf das Faktorisierungsproblem für univariate Polynome über R zu reduzieren.

- a) Sei die Abbildung $S_d : R[x_1, \dots, x_n] \rightarrow R[y]$ durch

$$h(x_1, \dots, x_n) \mapsto h(y, y^d, \dots, y^{d^{n-1}})$$

definiert (für $d \in \mathbb{N}$). Überzeugen Sie sich, dass S_d ein Homomorphismus ist, der für diejenigen Polynome invertiert werden kann, die in jeder Variablen einen Grad kleiner als d haben.

- b) Wir wollen mit S_d^{-1} die additive Abbildung $R[y]/\langle y^{d^n} \rangle \rightarrow R[x_1, \dots, x_n]$ bezeichnen, die

$$S_d^{-1}(c y^\alpha) = c x_1^{\alpha_1} \cdots x_n^{\alpha_n}$$

erfüllt, wobei $\alpha_1 + \alpha_2 d + \cdots + \alpha_n d^{n-1}$ die (positive) d -adische Darstellung von α sei.

Sei nun $f \in R[x_1, \dots, x_n]$, $d > \max_{1 \leq i \leq n} \deg_{x_i}(f)$, und sei g ein Faktor von f . Zeigen Sie, dass es dann irreduzible Faktoren g_1, \dots, g_s von $S_d(f)$ gibt, so dass $g = S_d^{-1}(\prod_{j=1}^s g_j)$ ist.

- c) Geben Sie nun einen Algorithmus an, der für ein $f \in R[x_1, \dots, x_n]$ seine irreduziblen Faktoren f_1, \dots, f_s berechnet. Untersuchen Sie die Laufzeit Ihres Algorithmus.
- d) Faktorisieren Sie das Polynom $f = -x^4y + x^3z + xz^2 + yz^2 \in \mathbb{F}_3[x, y, z]$ mit Ihrem Algorithmus.

zu **Aufgabe42:** Der Algorithmus sieht so aus:

Berechne Q , Nullraumbasis $v^{[1]}, \dots, v^{[r]}$ von $Q - I$.

factors $\leftarrow \{a(x)\}; r \leftarrow 2$

while |factors| $< r$

foreach $u(x) \in$ factors

foreach $s \in \text{GF}(q)$

$g(x) \leftarrow \text{ggT}(v^{[r]}(x) - s, u(x))$

if $g(x) \neq 1$ **or** $g(x) \neq u(x)$

 factors \leftarrow factors $\setminus \{u(x)\}$

$u(x) \leftarrow u(x)/g(x)$

 factors \leftarrow factors $\cup \{u(x), g(x)\}$

if |factors| = r **then return** factors

$r \leftarrow r + 1$

Man braucht nicht alle Linearkombinationen, weil die genannten ggTs schon genügen, um alle Faktoren zu unterscheiden. Für jedes paar von Faktoren $a_i(x)$ und $a_j(x)$ gibt es einen Basisvektor $v(x)$ von \mathcal{B} und $s_1 \neq s_2$, so dass $v(x) \equiv s_1 \pmod{a_i(x)}$, aber $v(x) \not\equiv s_2 \pmod{a_j(x)}$, sonst wäre $a(x)$ nicht quadratfrei. Also erhalten wir spätestens bei diesem Basispolynom zwei unterschiedliche Faktoren von a , so dass a_i Teiler des einen und a_j Teiler des anderen ist.

Da man zusätzlich noch garantieren kann, dass $v^{[1]}(x) = 1$, und bei *ggTs* damit höchstens ein konstanter Faktor abfallen würde, brauchen wir diesen Basisvektor nicht zu berücksichtigen.

zu **Aufgabe43:**

- a) S_d Ring-Homomorphismus ist klar. Sei U die Menge der Polynome, die in jeder Variablen Grad kleiner als d haben. U hat die Basis $\{x_1^{e_1} \cdots x_n^{e_n} \mid 0 \leq e_i < d\}$. Für ein $u \in U$ gilt $\deg S_d(u) \leq (d-1) \sum_{1 \leq i \leq n} d^{i-1} = d^n - 1$. $S_d(U)$ hat die Basis $\{x^i \mid 0 \leq i \leq d^n - 1\}$. S_d bildet die Basiselemente von U isomorph auf die von S_d ab, damit ist S_d auf ganz U isomorph.
- b) Es gilt $f \in U$. Da $g \mid f$, gilt $\deg g \leq \deg f$, d.h. $g \in U$. $S_d(g)$ liegt in einem ZPE-Ring und lässt sich zerlegen in ein c und geeignete g_i , so dass $c \prod_{1 \leq i \leq s} g_i$. Dann gilt $S_d^{-1}(c \prod_{1 \leq i \leq s} g_i) \mid f$.

Das heißt, für einen Faktor g von f gilt auch $S_d(g) \mid S_d(f)$, die Umkehrung gilt nicht. Wie man am Beispiel $x^2 + y^2 - 2 \in \mathbb{Z}_3[x, y]$, das irreduzibel ist, verfolgen kann. Für den Algorithmus nutzen wir natürlich die Faktorisierung im homomorphen Bild aus, ein ZPE-Ring. Man bekommt allerdings im Allgemeinen zu viele Faktoren, und muss durch Kombination dieser Faktoren die tatsächlichen wieder bestimmen. Das führt dann dazu, dass der Algorithmus nicht mehr polynomial sein kann, weil es bekanntlich irreduzible Polynome gibt, deren homomorphes Bild vollständig

zerfällt, man aber alle Kombinationen ausprobieren muss, um sich dessen sicher zu sein.

- c) Algorithmus: 1. S_d anwenden. 2. Im univariaten Ring faktorisieren: g_i, \dots, g_r . Betrachte alle Faktorkombinationen h der g_i , $h' = S_d(f)/h$ und prüfe, ob $f = S_d^{-1}(hh')$. Falls h ein Faktor ist, dann mit den übrigen Faktoren von h' weitermachen, ansonsten nächste Kombination.

Die Komplexität wird im wesentlichen durch $2^r M(d^n) n \log d$ beschrieben, wobei $r < d^n$, also ist der Algorithmus einfach exponentiell im Grad und doppelt exponentiell in der Anzahl der Variablen.

- d) Faktorisieren Sie das Polynom $f = -x^4y + x^3z + xz^2 + yz^2 \in \mathbb{F}_3[x, y, z]$ mit Ihrem Algorithmus.

Das Ergebnis ist: $2(x^2 + z)(x^2y + 2xz + 2yz)$. Es gilt:

$$\begin{aligned} S_d(2(x^2 + z)(x^2y + 2xz + 2yz)) &= (x^2 + x^{25})(2x^7 + x^{26} + x^{30}) \\ &= [x^2(x+1)(x^{11} + 2x^{10} + \dots)(x^{11} + x^8 + \dots)] \\ &\quad \cdot [x^7(x+1)^2(x^{16} + 2x^{14} + \dots)(x^5 + x^4 + \dots)] \end{aligned}$$

Der Algorithmus liefert:

$$\begin{aligned} S_3(f) &= -x^4x^5 + x^3x^{25} + xx^{50} + x^5x^{50} = -x^9 + x^{28} + x^{51} + x^{55} \\ &= x^9(x^5 + x^4 + x^3 + 2x^2 + x + 2)(x^{11} + 2x^8 + x^6 + x^4 + 2x^3 + x^2 + x + 1) \\ &\quad (x^{16} + 2x^{14} + x^{13} + 2x^{12} + x^{11} + 2x^9 + x^7 + 2x^6 + 2x^5 + 2x^3 + x^2 + 2x + 1) \\ &\quad (x^{11} + 2x^{10} + x^9 + x^8 + 2x^7 + x^5 + x^3 + 1)(x+1)^3 = \text{s.o.} \end{aligned}$$

Dabei wurde das bekannte Ergebnis und dessen Faktorisierung im homomorphen Bild in den letzten beiden Schritten schamlos ausgenutzt, um sofort die richtige Kombination zu raten. Es sind noch Verbesserungen möglich, mit denen man das Verifizieren im Fehlerfall verkürzen kann. Siehe Aufg. 16.16 vzG Modern CA.