Symbolisches Rechnen
Vorlesung
Wintersemester 2006, 2014
Sommersemester 2021, 2025

Johannes Waldmann, HTWK Leipzig

15. Mai 2025

Einleitung

Symbolisches Rechnen: Beispiele: Zahlen

- numerisches Rechnen mit Maschinenzahlen

```
sqrt 2 + sqrt 3  ==> 3.146264369941972 6
(sqrt 2 + sqrt 3)*(sqrt 2 - sqrt 3) ==> ...
```

- exaktes Rechnen (mit algebraischen Ausdrücken)

  $(\sqrt{2}+\sqrt{3})\cdot(\sqrt{2}-\sqrt{3}) = \ldots,$

  maxima: `expand(%)`

Symbolisches Rechnen: Beisp.: Funktionen

- auf konkreten Daten:

```
let f x = (x+1)^2 in f 3.1 - f 3
```

- auf symbolischen Daten: `diff((x+1)^2,x)`

- `subst([x=3],diff((x+1)^2,x))`

- eigentlich `diff(\x -> (x+1)^2)`

  mit `diff::(R -> R) -> (R -> R),`

Symbolisches Rechnen: Motivation

hat weitreichende Anwendungen:

- Lösen von (parametrisierten) Aufgaben*klassen*
  (für numerisches Rechnen muß Parameter fixiert werden)
- *exaktes* Lösen von Aufgaben
  (numer. R. mit Maschinenzahlen: nur Approximation)
- experimentelle, explorative, exakte Mathematik und Programmierung

ist nützlich im Studium, verwendet und vertieft:

- Mathematik (Analysis, Algebra)
- Algorithmen-Entwurf, -Analyse
- Prinzipien von Programmiersprachen

Überblick

- Zahlen (große, genaue)
- Vektoren (Gitterbasen)
- Polynome
- Terme, Term-Ersetzungs-Systeme
  (Anwendung: Differentiation, Vereinfachung)
- Termination un Vervollständigung v. Reduktionssystemen
  - für Polynom-Ideale, Anw.: Geometrische Beweise
    für Gleichheits-Theorien auf Termen (Knuth-Bendix)
- maschinell unterstütztes interaktives Programmieren und Beweisen in konstruktiver Typ-Theorie (Agda)

Literatur

- Wolfram Koepf: *Computeralgebra*, Springer, 2006. https://www.mathematik.uni-kassel.de/~koepf/CA/
- Hans-Gert Gräbe: *Einführung in das Symbolische Rechnen, Gröbnerbasen und Anwendungen*, Skripte, Universität Leipzig https://www.informatik.uni-leipzig.de/~graebe/skripte/
- Franz Baader and Tobias Nipkow: *Term Rewriting and All That*, Cambridge, 1998. https://www21.in.tum.de/~nipkow/TRaAT/
- weitere Literatur siehe z.B. https://portal.risc.jku.at/Members/hemmecke/teaching/ppscs

Software

- wir benutzen
  - Maxima `https://maxima.sourceforge.net/`
  - FriCAS `https://github.com/fricas/fricas/`
  - Geonext `https://geonext.uni-bayreuth.de/`
  - GHC `http://www.haskell.org/ghc/`
  - Agda `https://wiki.portal.chalmers.se/agda/pmwiki.php`
- ist alles im Pool installiert (ssh, tmux, x2go)
- allgemeine Hinweise, auch zum Selbstbauen `https://www.imn.htwk-leipzig.de/~waldmann/etc/cas/`

Beispiel: S.R. und Term-Ersetzung

Regeln für symbolisches Differenzieren (nach `t`):

```
D(t) -> 1                D(constant) -> 0
D(+(x,y)) -> +(D(x),D(y))
D(*(x,y)) -> +(*(y,D(x)),*(x,D(y)))
D(-(x,y)) -> -(D(x),D(y))
```

Robert Floyd 1967, zitiert in: Nachum Dershowitz: *33 Examples of Termination*,
`https://doi.org/10.1007/3-540-59340-3_2`

- Korrektheit? Termination? Komplexität?
- Strategie (Auswahl von Regel und Position)?
- ausreichend? angemessen?

Beispiel: Termersetzung (cont.)

```
data E = Zero | One | T
       | Plus E E | Times E E  deriving Show


e :: E
e = let b = Plus T One in Times b b


d :: E ->  E
d e = case e of
    Zero -> Zero ; One -> Zero ; T -> One
    Plus x y -> Plus (d x) (d y)
    Times x y ->
        Plus (Times y (d x)) (Times x (d y))
```

Beispiel: Inverse Symbolic Calculator

- `http://wayback.cecm.sfu.ca/projects/ISC/ISCmain.html` (kaputt)

  zur Bestimmung ganzzahliger Relationen (z.B. zwischen Potenzen einer numerisch gegebenen Zahl)

- `sqrt(2+sqrt 3)  ==>  1.9318516525781366`

```
integer relations algorithm, run:
K = 1.9318516525781366
                                   4       2
K satisfies the polynomial, X  - 4 X  + 1
```

  mit LLL-Algorithmus (Lenstra, Lenstra, and Lovasz, 1982), der kurzen Vektor in geeignetem Gitter bestimmt.

Hausaufgaben KW 15, Organisatorisches

1. zum Haskell-Programm zum Symb. Differenzieren:
   - füge Syntax und Regel für Quotienten hinzu
   - schlage Regeln zur Vereinfachung vor
2. ISC Simple Lookup and Browser sagt für $\sqrt{2+\sqrt{3}}$:

   ```
   Mixed constants with 5 operations
   1931851652578136 = 1/2/sin(Pi/12)
   ```

   begründen Sie das (geometrisch oder schriftlich)
3. ein Polynom mit Nullstelle $\sqrt[2]{2}+\sqrt[3]{3}$ bestimmen, nachrechnen.
4. Geonext: Satz von Napoleon illustrieren (gleichseitige Dreiecke über den Seiten eines beliebigen Dreiecks)

5. eigener Rechner: `rlwrap maxima` installieren,

   Rechner im Pool: `ssh` und `tmux` ausprobieren, auch Management von Sessions, Windows, Panes (split horizontal, vertikal), vgl. `https://news.ycombinator.com/item?id=26670708`

Organisatorisches:

- in Gitlab.Dit-Projekt einschreiben
- Hausaugabe: Wiki anmelden, Issue: diskutieren, ggf. MR
- Prüfungszulassung: Hausaufgaben, autotool (empfohlen, nicht erzwungen)
- Prüfung: Klausur, ggf. Hilfspunkte aus Projekt ($=$ ausgearbeitete Hausaufgabe o.ä.)

Zahlen

Überblick

- exakte Zahlen: natürlich, ganz, rational
  Darstellungen: Positionssystem, Bruch
  Rechnungen: Addition, Multiplikation
- beliebig genau genäherte Zahlen
  (berechenbare reelle Zahlen)
  Darstellungen: Positions-System, Kettenbruch
  Rechnungen: arithmetische und irrationale Funktionen
- später:
  exaktes Rechnen mit algebraischen Zahlen

Darstellung natürlicher Zahlen

- die Zahl $n \in \mathbb{N}$ im Positionssystem zur Basis B:

  $n = \sum_{k \geq 0} x_k B^k$

  mit $\forall i : 0 \leq x_i < B$ und $\{i \mid x_i \neq 0\}$ endlich

  Bsp: $25 = 1 \cdot 3^0 + 2 \cdot 3^1 + 2 \cdot 3^2 = [1, 2, 2]_3$

- Darstellung ist eindeutig, kann durch fortgesetzte Division mit Rest bestimmt werden $25 = 1 + 3 \cdot 8, 8 = 2 + 3 \cdot 2$

- praktisch wählt man die Basis
  - 10 für schriftliches Rechnen
  - historisch auch 60 (Zeit- und Winkelteilung)
  - 2 (oder 2^w) binäre Hardware, maschinennahe Software

Natürliche Zahlen, Addition

- Darstellung

```
type Digit = Word64 ; data N = Z | C Digit N
```

- Semantik

```
value :: N -> Natural
value Z = 0 ; value (C x xs) = x + 2^64 * value xs
```

- Rechnung

```
instance Num N where (+) = plus_carry False
plus_carry :: Bool -> N -> N -> N
plus_carry cin (C x xs) (C y ys) =
  let z = bool 0 1 cin + x + y
      cout = z < x || z < y || ...
  in  C z (plus_carry cout xs ys)
plus_carry ...
```

Rekursive Multiplikation

- anstatt „Schulmethode“: rekursive Multiplikation

  $(p + qB)(r + sB) = pr + (ps + qr)B + qsB^2$

  Bsp. $B = 100$, $(12 + 34 \cdot 100)(56 + 78 \cdot 100) = 12 \cdot 56 + \ldots$

  $(1 + 2 \cdot 10)(5 + 6 \cdot 10) = 1 \cdot 2 + \ldots$

- $M(w) =$ Anzahl elementarer Operationen (Plus, Mal) für Multiplikation w-stelliger Zahlen nach diesem Verfahren

  $M(1) = 1, M(w) = 4 \cdot M(w/2) + 2 \cdot w$

  $M(2^k) = 3 \cdot 4^k - 2^{k+1}$, also $M(w) \in \Theta(w^2)$

- also wie bei Schulmethode, aber das läßt sich verbessern, und darauf muß man erstmal kommen

Karatsuba-Multiplikation

- Anatoli Karatsuba, Juri Ofman 1962
- $(p + qB)(r + sB) = pr + (ps + qr)B + qsB^2$
  $= pr + ((p + q)(r + s) - pr - qs)B + qsB^2$
- $K(w) =$ Anzahl elementarer Operationen (Plus, Mal) für Multiplikation w-stelliger Zahlen nach diesem Verfahren
  $K(1) = 1, K(w) = 3 \cdot K(w/2) + 5 \cdot w$
  (eine Multiplikation weniger, drei Additionen mehr)
- asymptotisch mit Ansatz $K(w) \approx C \cdot w^e$
  $C \cdot w^e = 3Cw^e/2^e + 5w \Rightarrow 1 \approx 3/2^e$, also $e = \log_2 3 \approx 1.58$
- explizite Werte für $w = 2^k$,
  ab wann besser als Schulmethode?

Ganze Zahlen

- Darstellung: Bsp GMP (GNU Multi Precision Library)
  Natürliche Zahl (magnitude) mit Vorzeichen (sign)
  nicht Zweierkomplement, denn das ist nur für Rechnungen modulo 2^w sinnvoll
- GHC (`integer-gmp`)

```
data Integer = S Int | Jp BigNat | Jn BigNat
data BigNat = BN ByteArray
```

- NB: ganzzahlige Zahlbereiche in Haskell/GHC:
  - `Int` (ist immer die falsche Wahl)
  - Maschinenzahlen (mod 2^{64}): `Int64`, `Word64`,
  - beliebig große: `Integer`, `Numeric.Natural`

Rationale Zahlen

- rationale Zahl q ist Paar von ganzen Zahlen (z, n)

  ```
  data Q = Q Integer Integer
  ```

- normalisiert: $n \geq 1$ und $\gcd(|z|, |n|) = 1$
- normalisierte Darstellung ist eindeutig
- Vorteil:

  semantische Gleichheit (dieselbe Zahl wird bezeichnet) *ist* syntaktische Gleichheit (der Komponenten)
- Nachteil (?)

  nach jeder arithmetischen Operation normalisieren
- nicht normale Zahlen verhindern: Konstruktor `Q` nicht exportieren (sonst `let q = Q 8 12`)

Explosion der Stellenanzahl (Beispiel)

- Funktion $f : \mathbb{Q} \to \mathbb{Q} : x \mapsto x/2 + 1/x$
- $x_k = f^k(1)$, $x_0 = 1, x_1 = 3/2, x_2 = 17/12, x_3 = 577/408$
- Zähler (und Nenner) in jedem Schritt (wenigstens) quadriert, d.h., Stellenzahl verdoppelt (Bsp: x_{10})

  $p/q \mapsto p/(2q) + q/p = (p^2 + 2q^2)/(2pq)$
- das ist typisch für Folgen arithmetischer Op.,

  Normalisierung wirkt nur selten verkleinernd.
- die Folge x_k nähert $\sqrt{2}$ an: $(p_k/q_k)^2 - 2 = 1/q_k^2$

  Konvergenz ist quadratisch (jeder Schritt quadriert den Fehler, verdoppelt Anzahl gültiger Stellen)

Endliche und unendliche „Dezimal“brüche

- Darstellug von Zahlen in $\{x \mid 0 \leq x < 1\}$
  mit Basis $1/B$ und Ziffern $\in \{0 \ldots B-1\}$
- Bsp: $B = 10$, $0.314 = 0 \cdot B^0 + 3 \cdot B^{-1} + 1 \cdot B^{-2} + 4 \cdot B^{-3}$
- hier sind auch unendliche Ziffernfolgen sinnvoll,
  bezeichnet Limes der Werte der endlichen Partialfolgen
- Konversion

```
decimal :: Rational -> [ Nat ]
decimal x = let q = truncate (fromRational x)
            in  q : decimal (10 * (x-q))
```

  Anwendung: vorige Näherung von $\sqrt{2}$

Berechenbare reelle Zahlen

- beschrieben wird hier nicht die exakte (symbolische) Rechnung, sondern eine konvergente Näherung
- Def. reelle Zahl r mit $0 \leq r < 1$ heißt *berechenbar* (zur Basis B), wenn die Funktion $d : \mathbb{N} \to \{0, \ldots, B-1\}$, welche die Darstellung von x zur Basis $1/B$ bestimmt, berechenbar ist (z.B. durch eine Turingmaschine)
- jede rationale Zahl ist berechenbar

  (Beweis: Dezimalbruch ist endlich oder periodisch)
- $\sqrt{2}$ ist berechenbar (Beweis: vorige Folge x_k)
- Menge $\mathbb{B}$ der berechenbaren Zahlen ist abgeschlossen unter arithmetischen Operationen (und weiteren)

Potenzreihen, Exponentialfunktion

- Satz von Taylor: wenn f oft genug diff-bar,

  dann $f(x_0 + d) = \sum_{k=0}^{n-1} f^{(k)}(x_0) d^k / k! + \Delta_n$

  mit $\exists 0 \leq d' \leq d : \Delta_n = f^{(n)}(x_0 + d') d^n / n!$

- Bsp: $f(x) = \exp(x)$, dann $f = f' = f'' = f^{(3)} = \ldots$ und

  $\exp(0 + d) = 1 + d + d^2/2 + d^3/6 + \ldots$

- ```
  take 100 $ decimal
    $ sum $ take 100 $ scanl (/) (1::Rational) [1..]
      ==>   [2,7,1,8,2,8,1,8,2,8,4,5,9,0,4 ... ]
  ```

  Fehlerabschätzung? (reicht die zweite 100 für die erste?)

- $\exp(1) = \exp(1/2)^2$, diese Reihe konvergiert schneller
  ```

Potenzreihe für Wurzelfunktion

- Taylor-Reihe von $\sqrt{x}$ an der Stelle 1

  Ableitungen mit maxima:

  `diff(sqrt(x),x,5);` $105/32 \cdot x^{-9/2}$

  `diff(sqrt(x),x,6);` $-945/64 \cdot x^{-11/2}$

  Vermutung $f^{(n)}(1) = (-1)^{n+1} \cdot (2n-3)!! \cdot 2^{-n}$

- $\sqrt{1+d} = 1 + \sum_{k>0} (-1)^{k+1} \frac{(2k-3)!!}{2k!!} d^k$

  $= 1 + \frac{1}{2} \cdot d - \frac{1 \cdot 1}{2 \cdot 4} \cdot d^2 + \frac{1 \cdot 1 \cdot 3}{2 \cdot 4 \cdot 6} \cdot d^3 - \frac{1 \cdot 1 \cdot 3 \cdot 5}{2 \cdot 4 \cdot 6 \cdot 8} \cdot d^4 \pm \ldots$

- für $\sqrt{2}$: Berechnung als $\sqrt{1+1}$ konvergiert langsam

  besser: $\sqrt{2} = 7/5\sqrt{1+1/49}$

Logarithmen

Taylor-Entwicklung $\log(1+x) = x - x^2/2 + x^3/3 - \ldots$
konvergiert sehr langsam für $x = 1$, gar nicht für größere x.

J.R. Young, 1835, siehe v. Mangoldt/Knopp, Bd. 2, S. 127

$$\begin{aligned} a = \log(16/15) &= 4\log 2 - 1\log 3 - \log 5, \\ b = \log(25/24) &= \ldots \\ c = \log(81/80) &= \ldots \end{aligned}$$

Umstellung ergibt $\log 2 = 7a + 5b + 3c, \ldots$ Aufgaben:

- alle Koeffizienten ausrechnen
- wie genau sind die Werte für $\log 2$ usw., wenn man nur die erste Näherung benutzt, also $\log(1+x) \approx x$
- woher bekommt man geeignete Brüche (z. B. für $\log 7$)?

Pi

darüber gibt es ganze Bücher (Aufgabe: finde Beispiele)

Ansatz: Taylor-Entwicklung von $\arctan x$

$$\arctan x = x - x^3/3 + x^5/5 - \ldots$$

- betrachte $x = 1/5$, $\alpha = \arctan x$,
  bestimme $\tan(4\alpha)$ (ist nahe bei 1)
- bestimme $\beta = 4\alpha - \pi/4$
  und y mit $\beta = \arctan y$ (ist nahe bei 0)
- $\pi/4 = 4 \arctan x - \arctan y$

(J. Machin, 1706, 100 Stellen; W. Shanks, 1873, 707 St.)

Hausaufgaben

1. Multiplikation in GMP (GNU Multiprecision Library)
   `https://gmplib.org/manual/Multiplication-Algorithms`
   (a) Karatsuba-Rechnung ist dort etwas anders als hier auf der Folie, warum?
   (b) GHC verwendet GMP für den Typ `Integer`.
   Bestimmen Sie experimentell den Anstieg der Rechenzeit bei Verdopplung der Stellenzahl, z.B.

```
:set +s
x = 10^10^8 :: Integer
odd x -- damit x ausgewertet wird
odd ((x-1)*(x+1)) -- die eigentliche Messu
  True
```

```
   (3.73 secs, 166,159,792 bytes)
y = x*x -- hat doppelte Stellenzahl
```

Ist die Anzahl der Bytes plausibel?
Diskutieren Sie mögliche verkürzte Auswertungen für `odd` ... Kann GMP/GHC das?

(c) Zusatz: warum ist (oder erscheint) `(x+1)^2` schneller als `(x+1)*(x+1)`?

2. diskutieren (Zusatz: implementieren) Sie die Darstellung von ganzen Zahlen mit negativer Basis $B \leq -2$
(und nichtnegativen Ziffern $\in \{0, \ldots, |B| - 1\}$ wie bisher)
Bsp: $B = -2$,
$-3 = 1 \cdot B^0 + 0 \cdot B^1 + 1 \cdot B^2 + 1 \cdot B^3 = 1 + 0 + 4 - 8$

(a) Eindeutigkeit, Konstruktion
(b) Arithmetik (Nachfolger, Addition, Multiplikation)

3. Bestimmen Sie die Taylor-Reihe für den Arcustangens an der Stelle 0
   wie auf Folie *Potenzreihe für Wurzelfunktion*
   Bestimmen Sie damit $x = \arctan(1/2), y = \arctan(1/3)$ auf (z.B.) 20 Stellen.
   Begründen Sie $x + y = \pi/4$. Rechnen Sie den Wert für π aus und vergleichen Sie mit einer verläßlichen Quelle.
   Kann man π nach diesem Verfahren, aber mit anderen Parametern, besser bestimmen? (mehr Stellen bei gleichem Aufwand)

4. Bestimmen Sie die Taylor-Reihe für den (natürlichen) Logarithmus an der Stelle 1. Bestimmen Sie damit $a = \log(6/5), b = \log(9/8), c = \log(10/9)$ auf (z.B.) 100 Stellen und daraus $\log 2$ als eine Linearkombination.

5. wie bestimmt man $\sqrt{3}$, $\sqrt{5}$, $\sqrt[3]{2}$
   Hinweis: $\sqrt[3]{2}$ als Fixpunkt von $x \mapsto (2x + 2/x^2)/3$,
   diese Gewichte ergeben quadratische Konvergenz,
   ist äquivalent zu Bestimmung der Nullstelle von
   $f(x) = x^3 - 2$ nach Newton-Verfahren:
   $f'(x) = 3x^2, x \mapsto x - f(x)/f'(x) = \ldots$

6. Jerzy Karczmarczuk: *The Most Unreliable Technique in the World to compute pi*, 2003?
   `https://web.archive.org/web/20051017081559/http://users.info.unicaen.fr/~karczma/arpap/lazypi.ps.gz`

Automatische Differentiation

Motivation, Anwendung

- Optimierungs-Aufgaben, Bsp: Koeffizienten eines neuronalen Netzes (gegebener Topologie) zur bestmöglichen Approximation einer gegebenen Funktion
- oder Aufgaben, die sich als O-A formulieren lassen,
  Bsp: Lösung eines (nicht linearen) Ungleichungs-Systems $\bigwedge_i L_i(x) \geq R_i(x)$ für $x \in \mathbb{R}^n$
  äq: $0 \stackrel{?}{=} \min_x \sum_i \max(-(L_i(x) - R_i(x)), 0)$
- lösen durch Folge $x^{(k+1)} = x^{(k)} + \alpha \cdot d$, mit $\alpha \in \mathbb{R}, d \in \mathbb{R}^n$
  wobei Schritt-Richtung $d = -(\nabla f)(x^{(k)})$, Gradient $(\nabla f)x := [\partial f / \partial x_i \mid 1 \leq i \leq n] =$ Vektor der partiellen Ablt.

Verfahren zur Gradientenbestimmung

- Aufgabenstellung (Spezifikation):
  - gegeben: Vektor $x \in \mathbb{R}^d$, arithmetische Fkt. $f : \mathbb{R}^d \to \mathbb{R}$ als symbolischer Ausdruck (Term, DAG)
  - gesucht: $(\nabla f)(x)$, Wert des Gradienten an Stelle x
- verschiedene Lösungsverfahren:
  - *symbolische* Differentiation: Term für ∇f
    ... der kann aber sehr groß werden
  - *numerische* Diff.: berechnet $(f(x + h \cdot \vec{e_i}) - f(x))/h$
    ... anfällig für Rundungs- und Auslöschungs-Fehler
  - *automatische* Diff.: symbolische Rechng. auf DAG von f
- Baydin, Pearlmutter, Radul, Siskind: *Automatic differentiation in machine learning: a survey*, 2015–2018
  `https://arxiv.org/abs/1502.05767`

Automatische Differentiation (AD)

- Implementierung: repräsentiere Funktionswert *und* Gradient in *einem* Objekt, dessen Typ impl. `Num`

```
data N v z = N {abs :: z, lin :: M.Map v z}

var::v->z->N v z; var v z = N z (M.singleton v 1)

instance (Ord v, Num z) => Num (N v z) where
  fromInteger i = N (fromInteger i) M.empty
  N a1 l1+N a2 l2 = N(a1+a2)(M.unionWith (+) l1 l2)
```

- Anwendung: $\sqrt{2}$ durch $\arg\min f$ mit $fx = (x^2 - 2)^2$

```
f :: Num a => a -> a ; f x = (x^2 - 2)^2
f @(Fixed E6) 1.4  ==> 0.001600
f @(N () (Fixed E6) ) (var () 1.4) ==>
  N 0.001600 (fromList [((),-0.224000)])
```

Stochastischer Gradientenabstieg

- Bsp: gesucht ist Netz mit zwei Schichten für XOR.
- volles Optimierungsproblem: minimiere Fehlerquadratsumme über alle (4) Eingabe/Ausgabe-Paare (benutze Gradient dieser Fkt.)
- Variante: in jedem Schritt ein E/A-Paar zufällig auswählen
  - weniger Rechenaufwand pro Schritt
  - Ausbruch aus lokalen Minima (voller Gradient $= 0$), die global schlecht sind
- Variante: je Schritt: betrachte eine zufällige Teilmenge der Parameter als konstant (Gradient nur für die anderen)
  - weniger Rechenaufwand pro Schritt
- Details: siehe VL Numerik für Maschinelles Lernen

Anwendung: Matrix-Interpretationen

- das Wort-Ersetzungs-System $R = \{ab \to ba\}$ terminiert (jede Ableitung ist endlich),

  denn $i : a \mapsto \begin{pmatrix} 2 & 0 \\ 0 & 1 \end{pmatrix}, b \mapsto \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$

  ist zu R kompatible Interpretation in wohlfundiertes

  monotones Monoid $(\begin{pmatrix} \geq 1 & \geq 0 \\ \geq 0 & \geq 1 \end{pmatrix}, \circ, >_M)$

  mit $x >_M y := x_{1,2} > y_{1,2} \wedge \forall i, j : x_{i,j} \geq y_{i,j}$

  kompatibel: $i(ab) = \begin{pmatrix} 2 & 2 \\ 0 & 1 \end{pmatrix} >_M \begin{pmatrix} 2 & 1 \\ 0 & 1 \end{pmatrix} = i(ba)$

- solche Interpretation für $\{aa \to aba\}$, $\{a^2b^2 \to b^3a^3\}$?

Constraint-Lösen durch Optimieren

- gesucht sind $a, b \in \mathbb{R}^{3\times 3}$ mit $a, b \in M$, $a^2 >_M aba$
- Constraints für $2 \times 3 \times 3$ Unbekannte: $a_{i,j} \geq 0, , b_{i,j} \geq 0$
  ersetze durch Fkt. von unbeschränkten $x_{i,j} \in \mathbb{R}$
  Bsp.: $a_{i,j} = x_{i,j}^2$, $a_{i,j} = \max(0, x_{i,j})$, $a_{i,j} = \max(x_{i,j}, -x_{i,j})$
  AD funktioniert auch für stückweise Funktionen

  ```
  instance Ord (N v z) where ...
  ```
- 3×3 Constraints für Produkte: $(a^2)_{i,j} \geq (aba)_{i,j}$
  Ungl. $f \geq g$ realisieren durch Strafterm $\max(-(f-g), 0)$
  zu minimieren ist Summe der Strafen, min soll 0 sein
- Modellierung so, daß der Gradient nützlich ist
  (also Strafterm *nicht*: if $f < g$ then 1 else 0)
- Constraints (Ungl.) *exakt* erfüllen (in $\mathbb{Q}$, nicht `Double`)

Literatur und Software (funktionaler) AD

- Jerzy Karczmarczuk *Functional Differentiation of Computer Programs.* ICFP 1998 `https://dl.acm.org/doi/10.1145/289423.289442`
- Faustyna Krawiec, Simon Peyton Jones, Neel Krishnaswami, Tom Ellis, Richard A. Eisenberg, Andrew Fitzgibbon: *Provably correct, asymptotically efficient, higher-order reverse-mode automatic differentiation*, POPL 2022, `https://dl.acm.org/doi/10.1145/3498710`
  Implementierung: Mikolaj Konarski, `https://github.com/Mikolaj/horde-ad`
- Edward Kmett 2010–, `https://hackage.haskell.org/package/ad`

Aufgaben

1. (H. Rosenbrock 1960) ein Test für Abstiegsverfahren ist
   $f(x, y) = (x^2 - y)^2 + y^2/100$,
   Veranschaulichen Sie den Werteverlauf mit
   `rlwrap maxima`

   ```
   f : (x^2-y)^2 + y^2/100;
   plot3d(f,[x,-50,50],[y,-1000,3000]);
   ```

   wo ist das globale Minimum?

   in welche Richtung geht der (entgegengesetze) Gradient im Punkt $(20, 100)$,

   wo verläuft die Folge der Näherungswerte, wird das globale Minimum erreicht? (1. diskutieren, 2. ausprobieren)

2. für den Typ `N v z`: weitere Funktionen implementieren (Division, recip, sqrt, exp, log),

   den so bestimmten Wert des Gradienten mit numerischer Differentiation vergleichen

   diese Fkt. in einer Minimum-Bestimmung verwenden

3. in neuronalen Netzen wird oft die Funktion $S : x \mapsto 1/(1 + \exp(-x))$ verwendet (sie bildet $\mathbb{R}$ monoton steigend auf $[0, 1]$ ab)

   bestimmen Sie deren Ableitung symbolisch (zu Fuß oder Maxima), vergleichen mit AD-Implementierung (vorige Aufgabe)

4. (Fortsetzung) ein k-stelliges Neuron mit Gewichten $w_0, w_1, \ldots, w_k \in \mathbb{R}$ ist die Funktion $x \mapsto S(w_0 + \sum_i w_i x_i)$.

ein vollständig verbundenes Netz mit e Eingängen und n Neuronen: das Neuron i sieht alle Eingänge sowie die Ausgänge der Neuronen $1 \ldots i-1$.

Bestimmen Sie (mit stochastischem Gradientenabstieg mit AD) die Gewichte eines vollständigen Netzes (mit möglichst wenig Neuronen) für

- XOR über alle Eingänge
- Majoriät (falls s ungerade)

5. für Optimierungsverfahren 2. Ordnung benötigt man die Hesse-Matrix einer Funktion (diese enthält alle zweiten Ableitungen). Modellieren und berechnen Sie diese durch eine Erweiterung des Typs `data N v z`.

   Vergleichen Sie mit exakten Werten (Bsp. maxima: `hessian((x^2-y)^2,[x,y]);`)

6. (Forschungsaufgabe) Matrix-Interpretationen für Wortersetzungssysteme

- verbesserte Modellierung (der Zielfunktion) und Parameter (Schrittweite)
- stochastischer Abstieg (1. Ordnung)
- Abstiegsverfahren 2. Ord. (oder Quasi-Newton)

Testfälle

- (leicht) $a \to b$, $ab \to ba$, $ab \to bba$, $aa \to aba$,
- (schwerer) $a^2b^2 \to b^3a^3$, $\{a^2 \to bc, b^2 \to ac, c^2 \to ab\}$
  System mit 3 Regeln: es genügt $>_M$ für eine Regel, für die anderen $\geq_M$. Wie kann man diese Bedingung als Zielfunktion einer Optimierung realisieren? ganzzahlige Lösungen sind bekannt (gefunden durch Bit-Blasting) gibt es kleinere nicht-ganzzahlige?

Polynome

Motivation (I): Polynom-Interpretationen

- Anwendung: wfmA $A = (\mathbb{N}, >, [\cdot])$, jedes $[f]_A$ ist Polynom.
  Bsp: $[f](x, y) = x^2 \cdot y$ (Vorsicht), $[g](x, y) = x^2 + y$
  Bsp: kompatibel mit $g(g(x, y), z) \to g(x, g(y, z))$?
- es müssen Aussagen der Form $\forall x, y, z : P(x, y, z) > 0$ (automatisch) nachgewiesen werden
- welche Approximation in der autotool-Aufgabe dazu?
- Zeige $x, y \geq 0 \Rightarrow (x + y)/2 \geq \sqrt[2]{xy}$
  Ü: Zeige $x, y, z \geq 0 \Rightarrow (x + y + z)/3 \geq \sqrt[3]{xyz}$
  Ü: Hilberts 17. Problem (nichtneg. P, das kein SOS ist)

Motivation (II): Algebraische Zahlen

Bsp: gesucht ist Minimalpolynom für $y = \sqrt{2} + \sqrt[3]{3}$

	1	$\sqrt[3]{3}$	$\sqrt{2}$	$\sqrt{2}\sqrt[3]{3}$	$\sqrt[3]{3^2}$	$\sqrt{2}\sqrt[3]{3^2}$
y^0	1	0	0	0	0	0
y^1	0	1	1	0	0	0
y^2	2	0	0	2	1	0
y^3	3	6	2	0	0	3
y^4	4	3	12	8	12	0
y^5	60	20	4	15	3	20
y^6	17	90	120	24	60	18

die letzte Zeile ist linear abhängig von den vorigen (Dimension des Vektorraumes ist ≤ 6), ergibt Darstellung von y^6 als rationale Linearkombination von $y^0, \ldots, y^5$.

Motivation (III): Geometrische Örter

- Satz: in jedem Dreieck liegen
  - die Seitenmitten
  - die Höhenfußpunkte
  - die Mitten der oberen Höhenabschnitte

  auf einem Kreis (Feuerbach-Kreis, 9-Punkte-Kreis).

- Beweis durch symbolisches Rechnen:
  - Koordinaten der Ecken $(A_1, A_2), (B_1, B_2), (C_1, C_2)$
  - ex. Polynom P, so daß: (X_1, X_2) liegt auf Umkreis der Seitenmitten gdw $P(A_1, A_1, B_1, B_2, C_1, C_2, X_1, X_2) = 0$
  - (X_1, X_2) ist ein Höhenfußpunkt: $Q(\dots) = 0$
  - zu zeigen ist $Q(\dots) = 0 \Rightarrow P(\dots) = 0$
    das ist (später) eine Aussage über Polynom-Ideale

Semantik und Syntax von Polynomen

- Polynom (semantisch) als Funktion: $\lambda x.x^3 - 3x^2 + 3x - 1$
- Polynom (syntaktisch)
  als Menge von Monomen $\{1x^3, -3x^2, 3x^1, -1x^0\}$,
  als Folge von Koeffizienten $[1, -3, 3, -1]$
- symbolisches Rechnen mit Polynomen (elementare Arithmetik)
- als Grundlage für (später) Algorithmus von Buchberger zur Bestimmung von Gröberbasen
- Wdhlg. (Zahlen)bereiche (Gruppen, Ringe, Körper),
  Anwendung: Polynome in mehreren Variablen

Gruppen, Ringe, Körper

- Definitionen (Wiederholung)
  - Monoid: Addition, Null
  - Gruppe: wie Monoid, und Subtraktion
  - Ring: wie Gruppe und zusätzlich Multiplikation, Eins
  - Körper: wie Ring und zusätzlich Division
  - Vektorraum (ü. Körper): Gruppe, skalare Multiplikation

- Bsp: $\mathbb{N}, \mathbb{Z}, \mathbb{Z}/6\mathbb{Z}, \mathbb{Z}/7\mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{R}^2, \mathbb{C}, \mathbb{Z}[X]$

- zusätzliche Eigenschaften (Beispiele)
  - Ring heißt nullteilerfrei: $\forall a, b : a \neq 0 \wedge b \neq 0 \Rightarrow (ab) \neq 0$
  - Ring heißt euklidisch (bzgl. Funktion $|\cdot| : R \to \mathbb{N}$), wenn $\forall a, b \neq 0 : \exists q, r : a = q \cdot b + r$ mit $|r| < |b|$

Polynome in einer Variablen

- Repräsentation:
  - dicht: Folge (Data.Sequence) der Koeffizienten
  - dünn: endliche Abbildung (Data.Map)
    Schlüssel: Grad, Wert: Koeffizient
    normalisierte Darstellung: nur Koeffizienten $\neq 0$ werden repräsentiert (dann einfacher Test auf $P = 0$)
- Auswertung: mit Horner-Schema
- Addition
  - in dichter Darstellung: `Data.List.zipWith (+)`
  - in dünner Darstellung: `Data.Map.unionWith (+)`
- Multiplikation: naiv: quadratisch, mit FFT: $n \log n$

Polynom-Division (über $\mathbb{Q}$)

- Beispiel: $A = (x^5 - 1), B = (x^2 - 1),$ $A : B =$ Quotient $Q = x^3 + x$, Rest $R = x - 1$

- Spezifikation: $A : B = Q$ Rest R gdw. $A = B \cdot Q + R \wedge \deg R < \deg B$.

- Bezeichnung: $\deg P$ (Grad) = der höchste Exponent,

  $\deg(x^5 - 1) = 5, \deg(4) = 0, \deg(0) = -\infty$.

- Implementierung $A : B$: falls $\deg A \geq \deg B$, dann $(c_1 \cdot x^{e_1} = A$ (höchstes Monom), $(c_2 \cdot x^{e_2} = B$ (höchstes Monom), $q = (c_1/c_2)x^{e_1 - e_2)}$, $A' = A - qB$, $(R', Q') = A' : B$, Resultat ...

Polynome in mehreren Variablen

- das wird in den allermeisten Anwendungen gebraucht
- mögliche Repräsentationen für $\mathbb{Z}[X_1, X_2, \ldots]$
  - geschachtelt: $= (\mathbb{Z}[X_1])[X_2, \ldots]$
    als 1-Var-Polynom, Koeffizienten sind $(n-1)$-Var-Pol.
  - flach: als endliche Abbildung
    Schlüssel: Monom (Bsp: $X_2^8 X_4^5$), Wert: Koeffizient
    Monom: als endliche Abbildung
    Schlüssel: Variable (Bsp: X_2, X_4) Wert: Exponent
- die geschachtelte Darstellung ist elegant (und man muß nichts neu programmieren)

  kann aber unpraktisch sein (wenn man „von oben“ eine innere Variable benutzt)

Polynom-Darstellung als geordnete Liste

- Mult. von Polynom mit Zahl ist strukturerhaltend

  Struktur `M.Map Natural c` wird hier nicht benötigt, könnte auch `[(Natural,c)]` sein (Liste, Invariante: aufsteigend nach Schlüssel, besser: absteigend, wg. Zugriff auf head)

  dafür effiziente Addition, Multiplikation?

- Addition: wie `merge`. Dabei Normalisierung

- Multiplikation: wiederholte Addition mit Verschiebung

Listen-Darstellung f. mehr Var.

- dünne Darstellung ist

  ```
  type Mono v  = M.Map v Natural,
  type Poly v c = M.Map (Mono v) c
  ```

- jeweils `Map s w` ersetzen durch `[(s, w)]` mit
  Invariante: absteigend nach `s`
  effiziente Implementierung arithmetischer Op.?
- Addition von Polynomen (benutzt `merge`)
- Multiplikation von Polynomen: benutzt: von Monomen
  die Monom-Ordnung muß monoton sein bzgl. Multiplikation
  wenn $m_1 > m_2$, dann $m \cdot m_1 > m \cdot m_2$.
  Beispiele: lexikografisch; Exponentensumme, dann lex.

Ideale

- Def: in Ring $R = (M, 0, +, 1, \cdot)$: Menge $I \subseteq M$ heißt *Ideal*,
  - wenn abgeschlossen unter Summe (mit sich)
    $\forall x \in I, y \in I : (x + y) \in I$.
  - und abgeschlossen bzgl. Multiplikation mit M (mit allen)
    $\forall x \in I, y \in M : (xy) \in I$.
- Def: das von $B \subseteq M$ *erzeugte* Ideal, Notation $\langle B \rangle$,
  ist das kleinste (bzgl. Inklusion) Ideal I mit $B \subseteq I$
  Satz: $\mathrm{Ideal}(B) = \langle B \rangle = \{\sum c_i b_i \mid c_i \in M, b_i \in B\}$
  Bsp: in $\mathbb{Z}$: $\mathrm{Ideal}(\{8, 12\}) = \langle 8, 12 \rangle = ?$
  Bsp: in $\mathbb{Q}[X, Y]$: $\mathrm{Ideal}(1)$, $\mathrm{Ideal}(0)$, $\mathrm{Ideal}(\{X + Y, X - Y\})$
- Bsp: gilt $(X^5 - Y^2) \in \mathrm{Ideal}(\{X^2Y - 1, XY^2 - 1\})$?
  nächste VL: Algorithmus für Polynom-Ideal-Mitgliedschaft

Hausaufgaben

1. Bezeichnungen:

   arithmetisches Mittel $A(x_1, \ldots, x_n) = (\sum x_i)/n$,
   geometrisches Mittel $G(x_1, \ldots, x_n) = \sqrt[n]{\prod x_i)}$.

   Aufgabe: Stelle $A(x_1, \ldots, x_4)^4 - G(x_1, \ldots, x_4)^4$ als Summe von Quadraten (SOS) von Polynomen dar.

   Hinweis: $A(x_1, x_2)^2 - G(x_1, x_2)^2 = (x_1 - x_2)^2/4$ und
   $A(x_1, x_2, x_3, x_4) = A(A(x_1, x_2), A(x_3, x_4)) \geq$
   $A(G(x_1, x_2), G(x_3, x_4)) \geq G(G(x_1, x_2), G(x_3, x_4)) =$
   $G(x_1, x_2, x_3, x_4)$.

   Zusatz: $A(x_1, x_2, x_3)^3 - G(x_1, x_2, x_3)^3$ ist kein SOS, kann aber als Bruch mit SOS im Zähler und $(x + y + z)$ im Nenner geschrieben werden. Dazu die SOS-Darstellung

der A-G-Ungleichung für die 4 Werte $x_1, x_2, x_3, t = A(x_1, x_2, x_3)$ benutzen, denn $A(x_1, x_2, x_3, t) = t$ und $G(x_1, x_2, x_3, t)^4 = G(x_1, x_2, x_3)^3 \cdot t$.

Hintergrund: Bruce Reznick: *Some Concrete Aspects of Hilbert's 17th Problem*, 2000, `https://faculty.math.illinois.edu/~reznick/`.

2. das Minimalpolynom für $\sqrt{2} + \sqrt[3]{3}$ nach angegebenem Verfahren ausrechnen und überprüfen.

   Ähnlich für $\sqrt{3} + \sqrt{5}$ oder (z.B.) $\sqrt[5]{3} + \sqrt[5]{5}$

3. Das Polynom P für „liegt auf Umkreis der Seitenmitten“ angeben (oder ähnlicher geometrischer Ort im Dreieck).

   o.B.d.A $A_1 = A_2 = B_1 = 0$ annehmen.

   Warum wäre zusätzlich $C_2 = 0$ doch eine B.d.A.?

4. für die in VL angegebene Implementierung von Polynomen:

   eine anderen Koeffizientenbereich (als `Rational`) benutzen, z.B. `Complex Rational`

   nichttriviale Rechnungen durchführen (z.B. Division von $X^{pq} - 1$ durch $X^p - 1$), Ergebnis prüfen,

   Laufzeit messen, Ausführung profilieren, teure Funktionen feststellen, ggf. verbessern

   Vergleichen mit derselben Rechnung in einem richtigen Computer-Algebra-System (maxima, fricas). (Nicht irgendwo online, sondern lokal, damit man messen und vergleichen kann.)

5. für $P = X^2Y + XY^2 + Y^2, Q = XY - 1, R = Y^2 - 1$:
   ist $P \in \mathrm{Ideal}(Q, R)$?
   Hinweis: nein. Hinweis: Betrachten Sie Werte von P für die gemeinsamen Nullstellen von Q, R.

6. gilt $X^5 - Y^2 \in \mathrm{Ideal}(X^2Y - 1, XY^2 - 1)$?
   Welches Resultat liefert der Nullstellentest?

7. Zeigen Sie $\mathrm{Ideal}(X + XY, Y + XY, X^2, Y^2) = \mathrm{Ideal}(X, Y)$.

8. Entscheiden Sie $X^2 + 1 \in \mathrm{Ideal}(X + 1)$,
   $X^2 - y^2 \in \mathrm{Ideal}(X + Y)$, geben Sie Algorithmus für
   $P \in \mathrm{Ideal}(Q)$ an.

9. der arktische Halbring $\mathbb{A} = (\{-\infty\} \cup \mathbb{N}, \max, -\infty, +, 0)$.

   Ist $\deg$ homomorph (strukturerhaltend) von $(\mathbb{Z}[X], +, 0, \cdot, 1)$ nach $\mathbb{A}$?

0. Def: *Monom-Ordnung*: total, monoton bzgl. Multiplikation, terminierend.

   Implementieren und Eigenschaften überprüfen/beweisen für *graded reverse lexicographic*: Ordnen nach absteigender Exponentensumme, bei Gleichheit nach aufsteigender Exponentenfolge

   Bsp. (für $X > Y$) $X^3Y^2 > X^1Y^3$, $X^3Y^2 > X^4Y^1$.

   Welche Eigenschaft(en) gehen dabei ohne Gradierung verloren?

1. Für Monom-Ordnung $>$ bezeichnet mdeg(M) für Monome: den Exponentenvektor (in der Reihenfolge der Variablen).

   und für Polynome ($\neq 0$): den größten solchen Vektor (ist eindeutig, weil die Ordnung total ist).

   Bsp.: für $X > Y$ ist mdeg$(X^2Y^5) = [2, 5]$,

   für lex-Ordnung ist mdeg$(X^2Y^5 - X^8) = [8, 0]$.

   Ausprobieren, beweisen, ergänzen: für Polynome $f, g \neq 0$ ist mdeg$(f \cdot g) =$ mdeg$(f) +$ mdeg(g) (komponentenweise Addition)

   Welche Beziehung zw. mdeg$(f + g)$ und mdeg(f), mdeg(g)? Ggf. unter welchen Randbedingungen?

Gröbnerbasen

Begriff, Motivation

- Def: Polynom-Ideal-Element-Problem: gegeben Polynome $P, B_1, \ldots, B_n$, gefragt: $P \in \mathrm{Ideal}(\{B_1, \ldots, B_n\})$.
  Bsp: $X^5 - Y^2 \stackrel{?}{\in} \mathrm{Ideal}(X^2Y - 1, XY^2 - 1)$
- Beispiel-Anwendung: Beweis geometrischer Aussagen, die sich algebraisch formulieren lassen.
- Algorithmus von Bruno Buchberger (1965) konstruiert besondere Basis G mit $\mathrm{Ideal}(G) = \mathrm{Ideal}(\{B_1, \ldots, B_n\})$
- benannt nach seinem Doktorvater Wolfgang Gröbner, (`https://genealogy.math.ndsu.nodak.edu/id.php?id=18956`, C. F. Gauß $\rightarrow^*$ F. Klein $\rightarrow^*$ ...)

Literatur

- B. Buchberger: Ein algorithmisches Kriterium für die Lösbarkeit eines algebraischen Gleichungssystems (1970) `https://www3.risc.jku.at/people/buchberg/refereed_publications.html`

  (in 1970-00-00-A: Verweis auf Implementierung im „Formelcode der ZUSE Z 23“)

- F. Baader und T. Nipkow: Term Rewriting and all that (Kapitel 8), Cambridge Univ. Press 1998.

- H.-G. Gräbe: Gröbnerbasen und Anwendungen (Skript, Uni Leipzig); `http://www.informatik.uni-leipzig.de/~graebe/skripte/`

Reduktion von Polynomen

- Bezeichnungen: für $p \neq 0$ (Bsp: $p = 3X^2Y - 5XZ^2$)
  $H_>(p)$, der Kopf von p: das größte (bzgl. $>$) Monom (mit Koeffz.) (Bsp: $H_>(p) = 3X^2Y$)
  $R_>(p)$, der Rest von p, so daß $p = H(p) + R(p)$
- $p \to_f q$, falls p ein Monom $a \cdot m$ enthält, das durch $H(f)$ teilbar ist (exist. m' mit $m = H(f)m'$) und $q = p - am'f$.
- für endliche Menge F von Polynomen: $(\to_F) = \bigcup_{f \in F} (\to_f)$
- Bsp. $F = \{f_1, f_2\}$ mit $f_1 = x^2y - x^2$, $f_2 = xy^2 - y^2$.
  Ordnung $>$ auf Monomen: gradiert lexikografisch
  $x^2y^2 \to_{f_1} x^2y^2 - y \cdot f_1 = x^2y \to_{f_1} x^2y - 1 \cdot f_1 = x^2$
- Satz: $(p \to_F^* q) \Rightarrow (p - q) \in \mathrm{Ideal}(F)$.
  Folgerung: $p \to_F^* 0 \Rightarrow p \in \mathrm{Ideal}(F)$.

Eigenschaften der Redukton: Termination

- $p \to_f q$, falls p ein Monom $a \cdot m$ enthält, das durch $H(f)$ teilbar ist (exist. m' mit $m = H(f)m'$) und $q = p - am'f$.
- Bsp. $x^2y^2 \to_{f_1} x^2y^2 - y \cdot f_1 = x^2y \to_{f_1} x^2y - 1 \cdot f_1 = x^2$
- Satz: $\to_F$ terminiert. (mit $(\to_F) = \bigcup_{f \in F}(\to_f)$)
- Def: Relation $\to$ terminiert, wenn es keine unendliche Kette $e_0 \to e_1 \to e_2 \to \ldots$ gibt.
- Monom-Ordnung $>$ (Monotonie, Termination)
  daraus abgeleitete Ordnung $\gg$ auf Polynomen (als absteigende Folge von Monomen, Koeffz. ignorieren)
  Satz: $>$ ist Monom-Ordnung $\Rightarrow \gg$ terminiert
  Satz: $p \to_f q \implies p \gg q$, Folgerung: $\to_F$ terminiert.

Eigenschaften d. Reduktion: Konfluenz?

- $F = \{f_1, f_2\}$ mit $f_1 = x^2y - x^2, f_2 = xy^2 - y^2$.
  $x^2y^2 \to_F^* x^2$ und $x^2y^2 \to_{f_2} ? \to_{f_2} y^2$.
  x^2y^2 hat (wenigstens) zwei $\to_F$-Normalformen.
- Wir wollen (Existenz und) Eindeutigkeit von Nf. Dann $P \to_B^* N$ ausrechnen und $N = 0$ testen.
- Def. Relation $\to$ heißt *konfluent*, wenn
  $\forall a, b, c : (a \to^* b \wedge a \to^* c) \Rightarrow \exists d : b \to^* d \wedge c \to^* d$.
- Satz: wenn $\to$ terminiert und konfluent ist, dann besitzt jedes a genau eine Normalform
  Def: a ist $\to$-Normalform, falls $\neg\exists b : a \to b$.
- $\to_F$ ist nicht notwendig konfluent. BB-Algor.: berechnet $G \supseteq F$ mit $\mathrm{Ideal}(G) = \mathrm{Ideal}(F)$ und $\to_G$ konfluent.
  Def: solches G heißt *Gröbner-Basis* für $\mathrm{Ideal}(F)$.

S-Polynome

- Konstruktion von G für F durch Vervollständigung (Hinzufügen von Polynomen $s \in \mathrm{Ideal}(F)$)
- Def: $S(f_1, f_2) := f_1 \cdot m/H(f_1) - f_2 \cdot m/H(f_2)$.
  mit $m = \mathrm{lcm}(H(f_1), H(f_2))$ (kleinstes gemeinsames Vielf.)
  Satz: $S(f_1, f_2) \in \mathrm{Ideal}(F)$.
- Beispiel $F = \{f_1, f_2\}$ mit $f_1 = x^2y - x^2, f_2 = xy^2 - y^2$
  $S(f_1, f_2) = f_1 y - f_2 x = x^2y - xy^2$
- Satz (hier ohne Bew.) Wenn für alle
  $f_1, f_2 \in F : S(f_1, f_2) \to_F 0$, dann ist $\to_F$ konfluent.
- Beispiel: $x^2y - xy^2 \to_F x^2 - y^2 \not\to_F 0$.

Der Buchberger-Algorithmus

- Eing.: endl. Menge F von Polynomen, Monom-Ord. $>$
  Ausgabe: Gröbnerbasis G für $\mathrm{Ideal}(F)$.

  Start: $G_0 = F$. Dann für $i = 1, 2, \ldots$

  falls $\exists f_1, f_2 \in G_i : S(f_1, f_2) \rightarrow^{!}_{G_i} s_i \neq 0$, (Normalf. $\neq 0$)

  dann $G_{i+1} = G_i \cup \{s_i\}$, sonst Halt mit Resultat G_i

- Satz: Dieser Algorithmus ist total und korrekt.

  Beweis: partiell korrekt wg. $\forall i : \mathrm{Ideal}(G_i) = \mathrm{Ideal}(F)$,

  total (terminierend): falls unendl. Folge $s_0, s_1, \ldots$,

  dann existieren $i < j$ mit $H(s_i) \mid H(s_j)$ (warum?)

  das ist ein Widerspruch (zu welcher Eigenschaft?)

Buchberger-Alg., Beispiel

- (Fortsetzung, $F = \{f_1, f_2\}$ wie oben.)
- $G_0 = F$, dann $S(f_1, f_2) \to_F x^2 - y^2 = f_3$, also $G_1 = \{f_1, f_2, f_3\}$.
- Neue Paare $S(f_1, f_3) = f_1 - yf_3 = y^3 - y^2 =: f_4$ ist Normalform,
- $S(f_2, f_3) = xf_2 - y^2 f_3 = -xy^2 + y^4 \to_{f_2} y^4 - y^2 \to_{f_4} y^3 - y^2 \to_{f_4} 0$.
- ```
  load ("grobner");
  poly_grobner([x^2*y-x^2,x*y^2-y^2],[x,y]);
  ```

```
        2      2     2    2   2    2   3    2
[x  y - x , x y  - y , y  - x , y  - y ]
```
```

Buchberger-Alg., Eigenschaften

- Resultat (G-Basis) und Schrittzahl hängen von der gewählten Ordnung auf Monomen ab.
- Verbesserung: nicht nur die neuen s nach G_i reduzieren, sondern auch die G_i nach s. (siehe Aufgabe)
- Die Schrittzahl kann trotzdem doppelt exponentiell sein.
- Mit manchen Ordnungen geht es „of“ schneller,

  (empfohlen wird graded reverse lexikografisch)

  da gibt es aber nur Erfahrungswerte.

  Aufgabe: ausprobieren (wie Ordnung in maxima einstellen?)

Hausaufgaben

1. zur Ordnung auf Monomen:
   (a) unterschiedliche Ordnungen ((graded) (reverse) lexikografisch) selbst implementieren.
   `instance Ord (Mono v) where ...`
   (b) stimmt die abgeleitete Ordnung
   `newtype Mono v = M [(v, Natural)] deriving`
   mit einer dieser Ordnungen überein?

2. zur Ordnung auf Polynomen:
   für Polynome in Variablen $X > Y > Z$: geben Sie möglichst lange $\gg$-Ketten an, die bei $X^3 + Y^2Z$ beginnen
   - bzgl. der lexikografische Ordnung auf Monomen
   - bzgl. der grad-lexikografische Ordnung auf Monomen

3. Kann $S(f_1, f_2)$ Monome mit höherem Grad enthalten als f_1 und f_2?

4. bestimmen Sie nach dem Buchberger-Algorithmus eine Gröbner-Basis B für $I = \mathrm{Ideal}(X^2Y - 1, XY^2 - 1)$.
   (auf Papier, mit maxima, mit eigener Impl.)

5. Benutzen Sie dieses B, um $X^5 - Y^2 \in I$ zu entscheiden.
   (Papier, maxima (`poly_grobner_member`), eigen)

6. Geben Sie Polynome c_1, c_2 mit
   $X^5 - Y^2 = c_1(X^2Y - 1) + c_2(XY^2 - 1)$ an.
   Erweitern Sie dazu den Buchberger-Alg. und das Reduktionsverfahren $\rightarrow_B$.
   (zum Vergleich: Euklid bestimmt $\gcd(a, b)$, erweiterter Euklid bestimmt c, d mit $ac + bd = \gcd(a, b)$.)

7. Eine Gröbner-Basis B heißt *reduziert*, wenn
   $\forall b \in B : \neg\exists c : b \to_{B\setminus\{b\}} c$.
   (b kann nicht durch die jeweils anderen Basis-Elemente reduziert werden)
   Ist die Basis auf Folie *Buchberger-Bsp.* reduziert? Wenn nein, bestimmen sie eine reduzierte B'.
   Ergänzen Sie die Eigenbau-Implementierung.
   Probieren Sie verschiedene Monom-Ordnungen.
   Die reduzierte Gröbner-Basis eines Polynom-Ideals ist im wesentlichen eindeutig.

8. Arjen Cohen: *Gröbner Bases, an Introduction* (in: *Some Tapas of Computer Algebra*, Springer, 1999):
   it is very hard to provide a good explicit upper bound

on [die Laufzeit des Buchberger-Algorithmus], bad examples are known.

Finden Sie solche *bad examples*

(a) selbst ausdenken

(b) mit Suchmaschine (welche Testfälle werden in Publikationen verwendet, die verbesserte Algorithmen beschreiben)

(c) brute force

und messen Sie den Ressourcenverbrauch (maxima, Eigenbau)

Plan (vorläufig)

- (15) Einleitung, symbolisches Diff.
- (17) Automatisches Differenzieren
- (19) Polynome (Repräsent., Operationen) Polynomideale
- (20) Monom-Ordnungen, Gröbnerbasen (GB)
- (21) Anwendung GB in Geometrie
- (23) abstrakte Reduktionssysteme (ARS) (Termination, Konfluenz)
- (24) Term-Ersetzungs-Systeme (TRS) (Unifikation, Vervollständigung)
- (25) Curry-Howard-Isomorphie (Programm $=$ Beweis)
- (26) (daten)abhängige Typen
- (27) interaktives Beweisen
- (28) Zusammenfassung, Ausblick