Constructive Algebra in Functional Programming and Type Theory

Anders Mörtberg

Introduction

- Master thesis: Haskell implementation of constructive algebra
- Current work:
  - Bézout and GCD domains in type theory
  - Gauss elimination in Haskell and type theory
  - Smith normal form in Haskell

Haskell

- Functional
- Pure - Easier to reason about programs
- Lazy - Infinite datastructures
- Type classes

Type classes

```
class Eq a where
  (==) :: a → a → Bool

class Ring a where
  (<+>) :: a → a → a
  (<*>) :: a → a → a
  neg   :: a → a
  zero  :: a
  one   :: a
```

Specification

- Specify with computable boolean functions

```
propAddAssoc :: (Ring a, Eq a) ⇒ a → a → a → Bool
propAddAssoc x y z = (x <+> y) <+> z == x <+> (y <+> z)
```

- (Ring a, Eq a) ⇒ Means that the type a is a "discrete ring"
- Can be tested using software testing techniques

Example

```
type Z = Integer

instance Ring Z where
  (<*>) = (*)
  (<+>) = (+)
  neg   = negate
  one   = 1
  zero  = 0

> quickCheck (propAddAssoc :: Z -> Z -> Z -> Property)
+++ OK, passed 100 tests.
```

Linear algebra over a field

- Solving systems of linear equations

$$MX = 0 \qquad MX = A$$

- Gauss elimination

Coherent rings

- Generalize the notion of solving equations to finding generators of solutions over rings
- Given a vector M there exist a matrix L such that $ML = 0$ and

$$MX = 0 \quad \leftrightarrow \quad \exists Y.\ X = LY$$

Representation in Haskell

```
type Vector a = [a]
type Matrix a = [[a]]

class Ring a ⇒ Coherent a where
  solve :: Vector a → Matrix a

propCoherent :: (Coherent a, Eq a) ⇒ Vector a → Bool
propCoherent m = isSolution (solve m) m
```

Properties of coherent rings

Theorem

In a coherent ring it is possible to solve homogenous systems of equations

$$MX = 0$$

```
solveMxN :: Coherent a ⇒ Matrix a → Matrix a
```

Properties of coherent rings

Theorem

Let R be an integral domain and $I, J \subseteq R$ two f.g. ideals then

$$I \cap J \text{ f.g.} \Rightarrow R \text{ coherent}$$

```
type Ideal a = [a]

solveInt :: (Ideal a → Ideal a → (Ideal a,[[a]],[[a]]))
         → Vector a
         → Matrix a
```

Strongly discrete rings

- A ring is strongly discrete if we can decide ideal membership, i.e. we can solve

$$a_1 x_1 + \cdots + a_n x_n = b$$

```
class Ring a ⇒ StronglyDiscrete a where
  member :: a → Ideal a → Maybe [a]
```

Bézout domains

- Non-Noetherian analogue of principal ideal domains
- PID: Every ideal is principal
  - Quantification over all ideals
- Bézout domain: Every finitely generated ideal is principal
- Equivalent definition:

$$\forall a\, b. \exists g\, a_0\, b_0\, x\, y.\, a = g a_0 \wedge b = g b_0 \wedge a_0 x + b_0 y = 1$$

Bézout domains

Theorem

Every Bézout domain is coherent

Theorem

Every Bézout domain is strongly discrete iff division is explicit

Theorem

Every Euclidean domain is a Bézout domain. In particular $\mathbb{Z}$ *and* $k[x]$ *are Bézout domains*

Prüfer domains

- Non-Noetherian analogue of Dedekind domains
- Every f.g. ideal is invertible: Given a f.g. ideal I there exists (f.g.) J such that IJ is principal
- First order characterization:

$$\forall x\, y.\, \exists u\, v\, w.\, ux = vy \wedge (1 - u)y = wx$$

Theorem

Given f.g. ideal I and J, we can find generators of $I \cap J$

Examples of Prüfer domains

Theorem

Every Bézout domain is a Prüfer domain (compare: Every PID is a Dedekind domain)

Theorem

Let R be a Bézout domain and L an algebraic extension of its field of fractions K. The integral closure of R inside L is a Prüfer domain.

- $\mathbb{Z}[\sqrt{-5}]$
- $k[x, y]$ with $y^2 = 1 - x^4$

Current work

- Bézout and GCD domains in type theory
- Gauss elimination over field in Haskell
  - Formalized in type theory using SSReflect by Cyril Cohen
- Smith normal form in Haskell

GCD domains

- Non-Noetherian analogue of unique factorization domains
- GCD domain: Every pair of elements have a greatest common divisor

$$\forall a\, b. \exists g\, x\, y.\, a = gx \land b = gy \land \forall g'.\, g' \mid a \land g' \mid b \to g' \mid g$$

GCD domains in SSReflect

- Based of integral domain with decidable equality and explicit units
- In a GCD domain this give explicit divisibility

$$\forall a\, b.\, a \nmid b \vee \exists x.\, b = ax$$

GCD domains in SSReflect

Theorem

Every Bézout domain is a GCD domain

Theorem

Euclids lemma: If $a \mid bc$ *and* $gcd(a, b) = 1$ *then* $a \mid c$

Future work

- Gauss lemma
- If R is a GCD domain then $R[x]$ is also a GCD domain
- Implement Euclidean rings and prove that they are Bézout domains

Smith normal form

- Let A be a nonzero $m \times n$ matrix over a PID. There exists invertible $m \times m$ and $n \times n$ matrices S,T such that

$$SAT = \begin{pmatrix} \alpha_1 & 0 & 0 & & \cdots & & 0 \\ 0 & \alpha_2 & 0 & & \cdots & & 0 \\ 0 & 0 & \ddots & & & & 0 \\ & & & \alpha_r & & & \vdots \\ \vdots & & & & 0 & & \\ & & & & & \ddots & \\ 0 & & & \cdots & & & 0 \end{pmatrix}$$

  and $\alpha_i \mid \alpha_{i+1}$
- The α_i are called the invariant factors of the matrix

Representation in Haskell

```
data Matrix a = Cons a [a] [a] (Matrix a)
              | Empty
```

$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

```
ex :: Matrix Z
ex = Cons 1 [2] [3] (Cons 4 [] [] Empty)
```

Future work: Smith normal form in SSReflect

- Convert Haskell implementation to type theory
- Need constructive PIDs

Future work: Constructive PIDs

- Mines, Richman, Ruitenburg: Bézout domains such that if we have a sequence $u(n)$ with $u(n+1) \mid u(n)$ then there exists k such that $u(k) \mid u(k+1)$
- In type theory this can be represented as that the relation

$$R(a, b) := a \mid b \;\&\&\; not(b \mid a)$$

is well-founded

Questions?

This work has been partially funded by the FORMATH project, nr. 243847, of the FET program within the 7th Framework program of the European Commission