Functional Programming with Abstract Algebra,

timetraveler314
University of Genshin
timetraveler314@outlook.com

1.

1.1. Basic Structures

Definition 1.1.1 (Left) Semigroup Action

Let $\langle M, \oplus \rangle$ be a semigroup and S be a set. A (left) semigroup action of M on S is a function $\cdot$: $M \times S \to S$ such that for all $m, n \in M$ and $s \in S$,

$$(m \oplus n) \cdot s = m \cdot (n \cdot s)$$

Example 1.1.1

Consider the semigroup $\langle \mathbb{R}, + \rangle$ and the vector set $\mathbb{R}^2$. The function $\circlearrowleft: \mathbb{R} \times \mathbb{R}^2 \to \mathbb{R}^2$ defined by a counter-clockwise rotation of s by angle m is a semigroup action.

$$(\alpha + \beta) \circlearrowleft s = \alpha \circlearrowleft (\beta \circlearrowleft s)$$

Extending actions to monoids is straightforward by adding the requirement that

$$\varepsilon \cdot s = s.$$

1.2. Monoid Constructions

How to construct a monoid out of a given monoid, a type and the arrow constructor?

Definition 1.2.1 Pointwise Monoid

Let M be a monoid and S be a set. The pointwise monoid of M and S is the set of all functions $S \to M$ with the operation of pointwise multiplication.

$$(f \oplus g)(s) = f(s) \oplus g(s),$$
$$\varepsilon(s) = \varepsilon$$

How to upgrade a semigroup to a monoid?

Example 1.2.1 Free Monoid from Semigroup

Use `Maybe`. `Maybe M` is a monoid with `Nothing` as the identity element and `Just m` as the non-identity element.

1.2.1.1. Cayley Representation

```
rep :: [a] -> ([a] -> [a])
rep xs = (xs ++)
```

```
abs :: ([a] -> [a]) -> [a]
abs f = f []
```

$$\langle [A], {+\!+}, [] \rangle \leftrightarrow \langle [A] \to [A], \circ, \mathrm{id} \rangle$$

The idea works for any monoid.

> **Definition 1.2.1.1.1** Cayley Representation
>
> Let M be a monoid. The Cayley representation of M is the set of all functions $M \to M$ with the operation of function composition.
>
> $$\begin{aligned}(f \circ g)(m) &= f(g(m)),\\ \mathrm{id}(m) &= m\end{aligned}$$

> **Definition 1.2.1.1.2** Free Monoid
>
> Let S be a set. The free monoid generated by S is the set of all finite sequences of elements of S with the operation of concatenation.

2. Day 2 : Automatic Differentiation

2.1. Structures

2.1.1. Semiring Module ("Scalar Product")

```
class (Semiring d, Monoid e)
    => Module d e | e -> d where
    (.*.) :: d -> e -> e
```

3. Day 3 : PLP/Graded Monoids