Algebraic Structure, Computational Benefits

Graham Enos

Februrary 2018

PR Reviews in DC

Me: “Hey Ben, you could make this a semigroup ... ooo no wait, it’s a monoid! I wonder if it commutes?”

Ben: (rolls eyes) “Sure Graham, whatever.”

Adam: (quits)

Nerdery

But with a purpose

Making algebraic structure explicit can yield computational benefits.

Semigroup

$(S, \oplus)$ forms a semigroup if $\oplus$ is associative:

$$\forall x, y, z \in S$$
$$(x \oplus y) \oplus z = x \oplus (y \oplus z)$$

In Scala, stolen from `cats`:

```
trait Semigroup[A] {
  def combine(x: A, y: A): A // aka |+|
}
```

Examples

- Integers with addition
- Doubles ($\mathbb{R}$) with maximum
- (Nonempty-) Lists with concatenation
- Square nonnegative matrices with multiplication

Monoid

$(M, \oplus)$ forms a monoid if it's a semigroup and there's an identity:

$$\exists \epsilon \in M \text{ such that } \forall x \in M$$

$$\epsilon \oplus x = x = x \oplus \epsilon$$

```
trait Monoid[A] extends Semigroup[A] {
  def empty: A
}
```

Examples

- Integers with addition and zero
- Doubles with maximum and $-\infty$
- Lists with concatenation and `[]`
- Square nonnegative matrices with multiplication and I

Why do we care

From Mathematics to Generic Programming

- big integer libraries: exponentiation
- matrices: scalar multiplication
- matrices: exponentiation (e.g. Fibonacci numbers)
- cryptography: double-and-add for elliptic curves
- cryptography: square-and-multiply for modular exponentiation

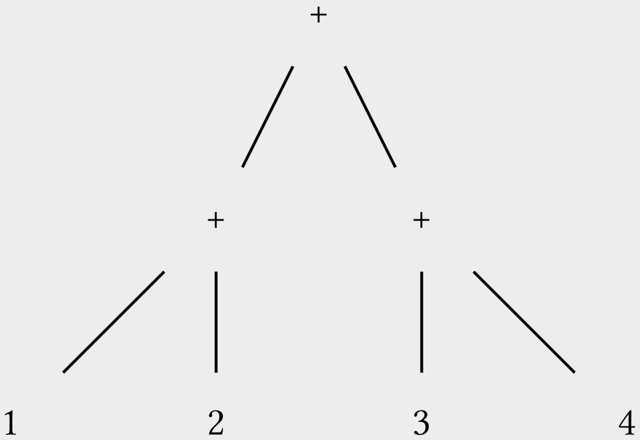
… all the same algorithm
"Egyptian multiplication," "Russian Peasant," etc.
logarithmic time instead of linear

More interestingly

Associativity $\Longrightarrow$ Parallelism

Parallel Summation

$$1 + 2 + 3 + 4 = (1 + 2) + (3 + 4)$$



Demo

`GaussianMonoid.sc` (lifted from `HLearn`)

I ♥ `foldMap`

From `cats.Foldable`:

```
def foldMap[A, B](fa: F[A])   // foldable collection
                 (f: A => B)  // mapping to a monoid
                 (implicit B: Monoid[B]): B =
  foldLeft(fa, B.empty)((b, a) => B.combine(b, f(a)))
```

You bring the mapping, I'll bring the aggregation `com.qf.stats.nlp.ner.evaluation.AylienNERTesting`

I needed a `Map[String, BinaryClassifierPerformance]`.
For each line of deserialized JSON:

1. extract text attribute
2. run 4 different NER models over the text
3. compare each result to expected, put into a map
4. aggregate all the maps for each model

I was going to `flatMap` and `eagerMapValues` and …

Writing that code

`foldMap` to the rescue

  monoid instance for `BinaryClassifierPerformance`

∴ `Map[String, BinaryClassifierPerformance]` a monoid

∴ code becomes a simple `foldMap`

Instances beget instances

In `cats`:

- `Either`, `List`, `Queue`, `Stream`, `String`, `Vector`, etc.
- `Monoid[A]`, `Monoid[B]` $\implies$ `Monoid[(A, B)]`
- `Semigroup[A]` $\implies$ `Monoid[Option[A]]`, `Semigroup[Future[A]]`, `Monoid[Map[K, A]]`

In `algebird`, an aggregation system where monoids are used for approximate data types (Bloom filter, Count-min sketch, HyperLogLog, etc.), moving averages, Gaussian distributions, etc.

Bonus round thanks to `tesser`

$(C, \oplus)$ a commutative monoid if a monoid and

$$\forall x, y \in C$$
$$x \oplus y = y \oplus x$$

Associativity $\implies$ `Map-Reduce`
Associativity + Commutativity $\implies$ `Map-Shuffle-Reduce`

Lord of the Semirings

$(S, \oplus, \otimes)$ is a semiring if

- $(S, \oplus)$ a commutative monoid
- $(S, \otimes)$ a monoid
- distribution:

$$x \otimes (y \oplus z) = (x \otimes y) \oplus (x \otimes z)$$
$$(x \oplus y) \otimes z = (x \otimes z) \oplus (y \otimes z)$$

- absorption: $x \otimes 0 = 0 = 0 \otimes x$

One of my favorite things

Tropical Semiring $(\mathbb{T}, \oplus, \otimes)$, where

$$\begin{aligned} \mathbb{T} &\coloneqq \mathbb{R} \cup \{\infty\} \\ x \oplus y &\coloneqq \min(x, y) \\ x \otimes y &\coloneqq x + y \end{aligned}$$

Shortest distance in graph theory $\mapsto$ matrix multiplication

`com.twitter.algebird.MinPlus`

Thanks!

Bibliography

- gratisography.com
- typelevel.org/cats
- underscore.io/books/scala-with-cats
- twitter.github.io/algebird
- izbicki.me
- github.com/aphyr/tesser
- *Nine Chapters on the Semigroup Art*, Cain
- *From Mathematics to Generic Programming*, Stepanov
- *Tropical Semirings*, Pin
- *Provenance Semirings*, Green et. al.
- ... `Haskell`