Tagless Final

Evolution Gaming's Scala Bootcamp
Raitis Veinbahs
2023.03.23.

Algebraic Data Types and Code

- Is this data?
- Is this code?

```
[1, 2, 3]
```

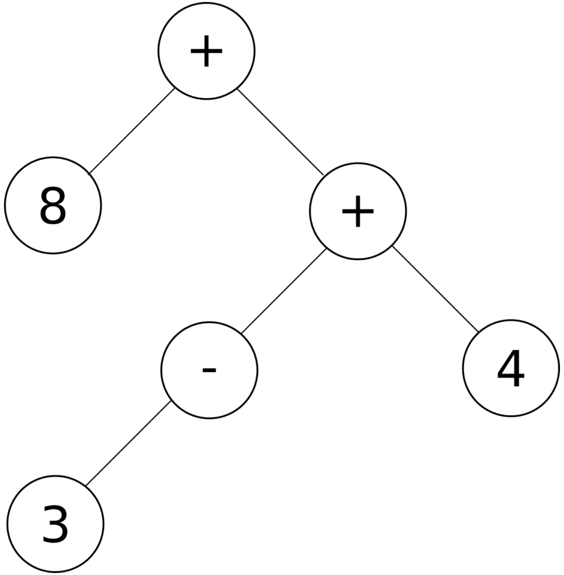
or

```
[].append(1).append(2).append(3)
```

Algebraic Data Types and Code

- Is this code?
- Is this data?

```
8 + ((-3) + 4)
```



Algebraic Data Types and Code

- Algebraic Data Types are data,
- but data can also be code (abstract) syntax trees.

- If ADTs can be viewed as code, then code analogies should make sense
- Folding ADT can be viewed as Interpreting Code

- Algebraic Data Types represent an algebra — set of operations on some set

Expression problem

Code

8 + (-3) + 4

Many meanings, four programs, but repeating code

Compute: = 9

Pretty print: “8 + (-3) + 4”

Counting marbles: List(o, o, o, o, o, o, o, o, o)

Others: both, colorful marbles, endless other options

Expression problem, the solution

Code

8 + (-3) + 4

Common part

For data of some type `repr`

```
number(n: int): repr // it’s still here, but it’s  embedded

negate(a: repr): repr

add(a: repr, b: repr): repr
```

Practical uses

- Ability to see more precise requirements (unless Sync)
- Mostly combining effect types
  - Id
  - StateT
  - IO (random, network, disk access)
- Number example in the real world
  - https://hackage.haskell.org/package/base-4.18.0.0/docs/GHC-Num.html#t:Num

References

https://okmij.org/ftp/tagless-final/course/lecture.pdf

https://okmij.org/ftp/tagless-final/course/Boehm-Berarducci.html

https://www.reddit.com/r/haskell/comments/nmj8hz/final_tagless_encodings_have_little_to_do_with/