Zipping Lists with Repetition

-- a puzzle

Koen Lindström Claessen

```
type Nat = Int
```

```
data List a
  = Unit a
  | List a :++: List a
  | Rep Nat (List a)
```

same **length**

```
zip :: List a -> List a -> List (a,b)
```

minimal **size**

```
len :: List a -> Integer
len (Unit x)   = 1
len (p :++: q) = len p + len q
len (Rep n p)  = n * len p
```

```
size :: List a -> Integer
size (Unit x)   = 1
size (p :++: q) = size p + size q
size (Rep _ p)  = size p
```

```
zip :: List a -> List a -> List (a,b)
```

polynomial in the size of the **result** (?)

independent of the **number** of repetitions

best **heuristic**

```
type List a = [Node a]

data Node a
  = Unit a
  | Rep Nat (List a)
```

https://github.com/koengit/zippuzzle