

Enumerating the Elements of the Eisenstein Array

Roland Backhouse

rcb@cs.nott.ac.uk

João F. Ferreira

joao@joaoff.com

March 2, 2009

Abstract

In [2], we discuss several algorithms that enumerate the elements of the Eisenstein Array [4]. In this document we show and discuss several Haskell implementations of these algorithms.

1 The Eisenstein Array

Given two natural numbers m and n , Stern [4] describes a process (which he attributes to Eisenstein) of generating an infinite sequence of rows of numbers. The *zeroth* row in the sequence (“nullte Entwicklungsreihe”) is the given pair of numbers:

$$m \quad n \quad .$$

Subsequent rows are obtained by inserting between every pair of numbers the sum of the numbers. Thus the *first* row is

$$m \quad m+n \quad n$$

and the *second* row is

$$m \quad 2 \times m + n \quad m+n \quad m+2 \times n \quad n \quad .$$

The process of constructing such rows is repeated indefinitely. The sequence of numbers obtained by concatenating the individual rows in order is what is now called the *Eisenstein array* and denoted by $Ei(m,n)$ (see, for example, [3, sequence A064881]). Stern refers to each occurrence of a number in rows other than the zeroth row as either a *sum element* (“Summenglied”) or a *source element* (“Stammglied”). The sum elements are the newly added numbers. For example, in the first row the number $m+n$ is a sum element; in the second row the number $m+n$ is a source element.

2 Newman's Algorithm

An interesting question is whether Stern also documents the algorithm currently attributed to Moshe Newman for enumerating the elements of $Ei(1,1)$ [2, section 4.2.3]. Newman's algorithm predicts that each triple of numbers in a given row of $Ei(1,1)$ has the form

$$a \quad b \quad (2 \left\lfloor \frac{a}{b} \right\rfloor + 1) \times b - a \quad .$$

In [1, appendix A], we have implemented Newman's algorithm as follows.

```
cwnEnum :: [Rational]
cwnEnum = iterate nextCW 1/1
  where nextCW :: Rational → Rational
        nextCW r = let (n, m) = (numerator r, denominator r)
                      j       = ⌊n/m⌋
                      in m / ((2×j + 1)×m - n)
```

For the purpose of this document, we are interested in the elements of $Ei(1,1)$, i.e., in the sequence of numerators given by *cwnEnum*. Function *newman* enumerates the elements of $Ei(1,1)$, using the infinite list created by *cwnEnum* (we have to detect a change in level).

```
newman :: [Integer]
newman = concatMap dlevel cwnEnum
  where dlevel r | (denominator r) == 1 = [numerator r, 1]
               | otherwise               = [numerator r]
```

3 Enumerating the Elements of $Ei(m,n)$

One way of enumerating the elements of the array $Ei(m,n)$ is:

```
ei :: Integer → Integer → [Integer]
ei m n = m : eiloop 1 1 m n m n
  where eiloop a 1 m n cm cn = n : cm : eiloop 1 (a + 1) cm (a×cm + cn) cm cn
        eiloop a b m n cm cn = let k = 2×⌊a/b⌋ + 1
                                in n : eiloop b (k×b - a) n (k×n - m) cm cn
```

We can test if the function *newman* is in fact enumerating the elements of $Ei(1,1)$. Let's compare the first 1000 elements of both enumerations:

```
>> (take 1000 newman) == (take 1000 (ei 1 1))
True
```

```
>> (take 1000 (map (2*) newman)) == (take 1000 (ei 2 2))
True
```

The part after the prompt, `>>`, is the Haskell code that `ghci` is executing. The result is shown in the subsequent line. The second command shows an instance of the property:

$$\text{map } (k \times) \text{ newman} == \text{ei } k \ k \quad .$$

References

- [1] Roland Backhouse and João F. Ferreira. Recounting the rationals: Twice! volume 5133, pages 79–91, 2008.
- [2] Roland Backhouse and João F. Ferreira. On Euclid’s algorithm and elementary number theory. 2009.
- [3] Neil J. A. Sloane. The On-Line Encyclopedia of Integer Sequences. <http://www.research.att.com/~njas/sequences/>.
- [4] Moritz A. Stern. Über eine zahlentheoretische Funktion. *Journal für die reine und angewandte Mathematik*, 55:193–220, 1858.