Universität der Bundeswehr München

Bachelorarbeit

Darstellung rationaler Zahlen durch Ägyptische Brüche

Eine Untersuchung von Algorithmen und Aufwand

Eingereicht von: Lars Berger
1173278

Aufgabensteller: Prof. Dr. Cornelius Greither

Betreuer: Dr. Soeren Kleine

Abgabedatum: 19.11.2019

Universität der Bundeswehr München
Fakultät für Informatik
Institut für Mathematik und Operations Research

Inhaltsverzeichnis

0 TODOs

Notes

1 Einleitung

Obwohl die Divergenz der Harmonischen Reihe zeigt, dass man mit Brüchen solcher Art durchaus alle Rationalen Zahlen $x \in \mathbb{Q}$ erzeugen kann, waren für ganzzahlige Werte in Ägypten Schreibweisen gängig, weshalb hier auf eine Betrachtung von Brüchen $\frac{a}{b} \geq 1$ verzichtet wird.

Folgende Algorithmen werden Betrachtet:

- Greedy-Algorithmus (auch Fibonacci-Sylvester-Algo)

    - Varianten (odd, even)

- Continued-Fraction Algorithm (Bleicher)
- Erdös
- Bleicher (?)

Fragen

- woher weiß ich, dass $(ax - b) < a$?
  immerhin gilt lediglich $x \geq 2$ und $a < b$ n.V., also müsste $2 \leq x < 1 + \frac{b}{a}$

2 Der Greedy Algorithmus

Eine der bekanntesten Methoden, eine Ägyptische Brucherweiterung für Brüche $\frac{a}{b}$; $a, b \in \mathbb{Q}$ zu finden, ist der Greedy Algorithmus. Dabei werden jeweils die größtmöglichen Stammbrüche $\frac{1}{x_i}$ gesucht, wobei

$$\frac{1}{x_i} \leq \frac{a}{b} - \sum_{j=1}^{i-1} \frac{1}{x_j} < \frac{1}{x_i - 1}, \tag{1}$$

wobei gilt, dass

$$x_i \neq x_j; \forall i \neq j = (1, .., i)$$

solange, bis

this is right, isn't it?

$$\frac{a}{b} = \frac{1}{x_1} + \frac{1}{x_2} + ... + \frac{1}{x_i} = \sum_{j=1}^{i} \frac{1}{x_j}.$$

Da in jedem Fall der größtmögliche, noch nicht vorhandene Bruch gesucht wird, der noch in die Summe der Stammbrüche passt, ohne dass diese zu groß wird, kann es zu sehr ungünstigen Ergebnissen mit extrem langen Divisoren kommen; ein anschauliches Beispiel dafür ist:

$$\frac{5}{121} = \frac{1}{25} + \frac{1}{757} + \frac{1}{763309} + \frac{1}{873960180913} + \frac{1}{1527612795642093418846225},$$

wobei man den Bruch auch folgendermaßen zerlegen kann:

$$\frac{5}{121} = \frac{1}{33} + \frac{1}{121} + \frac{1}{363}.$$

Durch diese Komplexitätsprobleme scheint es unsinnig, den Greedy-Algorithmus zu verwenden. Trotz hat WERAUCHIMMER IRGENDWANN nachgewiesen, dass der Algorithmus tatsächlich terminiert. Im Anhang 3.1 findet sich eine eigene Implementierung des Greedy-Algorithmus.

Namen und Quelle finden

wann?

Beweis einfü

Satz 2.1. *Der Greedy-Algortihmus terminiert für paarweise verschiedene Brüche* $\frac{1}{x}$

Beweis. Für die erste Iteration des Greedy-Algorithmus ergibt sich aus 1:

$$\frac{1}{x} \leq \frac{a}{b} < \frac{1}{x-1}$$

Daraus folgt ein Rest r von

$$r = \frac{a}{b} - \frac{1}{x} = \frac{ax - b}{bx},$$

der den Zähler $(ax - b)$ hat, der kleiner als a ist.

woher weiß i das?

Somit verkleinert sich dieser Rest mit jedem Schritt und erreicht irgendwann Null, wie gefordert. □

3 Anhang

3.1 PARI Code für den Greedy-Algorithmus

```
listsum( list ) = sum(i=1,#list,list[i]);


fibonacci_sylvester(fraction, stepsize, start)=
{ local(candidate, result);
   result=List();
  candidate = start;

  /* print error if fraction is larger one */
  if (numerator(fraction) > denominator(fraction),
    print("fraction is larger than 1. Use fractions smaller or equal to 1.");
    return;
  );

  /* check if fraction is a unit fraction */
  if (numerator(fraction) == 1,
    print("fraction was already a unit fraction");
    listput(result, fraction);
  );

  /* calculate summands and add them to the result */
  while (listsum(result) < fraction,
    candidate += stepsize;
    while (1/candidate > fraction - listsum(result),
      candidate += stepsize;
    );
    print("adding ", 1/candidate);
    listput(result, 1/candidate);
  );

  return(result);
}


greedy(fraction) = fibonacci_sylvester(fraction, 1, 1);
greedy_odd(fraction) = {print("\nthis might not come to an end!\n"); fibonacci_sylvester(fraction, 2, 1);}
greedy_even(fraction) = fibonacci_sylvester(fraction, 2, 0);
```

Eidesstattliche Erklärung

Hiermit versichere ich, dass die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt wurden.
Ferner habe ich vom Merkblatt über die Verwendung von Bachelor/Masterabschlussarbeiten Kenntnis genommen und räume das einfache Nutzungsrecht an meiner Bachelorarbeit der Universität der Bundeswehr München ein.

Neubiberg, den 19.11.2019

Lars Berger