Vergleich verschiedener Varianten von ggT Algorithmen von großen Zahlen Zusammenfassung

12. Januar 2020 - 19:41

Licina Esada
Universität von Luxemburg
Email: esada.licina.001@student.uni.lu

Volker Müller
Universität von Luxemburg
Email: volker.müller@uni.lu

***Abstrakt*—Diese Zusammenfassung über den finalen Bericht,wurde von der Studentin Esada Licina und mit der Unterstützung ihres Projekt Tutors Volker Müller gemacht.In diesem Projekt wird der wissenschaftliche und technische Teil beschrieben.Für mehr Information,bitte den finalen Bericht durchlesen.**

1. Einführung und Projektbeschreibung

Die Menschheit hat vor Jahren angefangen Wege zu suchen um Mathe Probleme zu lösen.Einer von denen ist der ggT Algorithmus.Das Word ggT steht für größter gemeinsamer Teiler und Algorithmus ist ein Weg um Probleme zu lösen in verschiedenen Stufen.Das Ziel ist verschiedene Varianten von ggT Algorithmen zu verglichen und zu beobachten welche von denen die höchste Leistung besitzt.Es sind im Ganzen 5 Algorithmen, der Euklidischer, der erweiterter Euklidischer, der Binärer, der erweiterter Binärer und der von Java geschriebene BigInteger Algorithmus.Hier wird mit der Programmiersprache Java gearbeitet und NetBeans erleichter die Erstellung von Java Swing Applikation. In Java werden die Algorithmen programmiert und in NetBeans wird ein virtuelles Fenster erstellt mit dem Resultaten.Excel ist da für ein Diagramm zu erstellen nachdem die Daten von NetBeans in Excel gespeichert wurden.

2. Voraussetzungen

Bevor man mit dem Projekt anfängt soll man Informationen über die verschiedenen ggT Algorithmen, NetBeans, Java und Excel sammeln.Dann soll man auch Erfahrung mit programmieren haben und Mathe Kenntnisse besitzen.

3. Wissenschaftliche Präsentation vom Vergleich der Leistung verschiedener ggT Algorithmen

3.1. Anforderungen

Funktionale Anforderungen:

1) Berechnung von der Zeit.
2) Zeigt die Dauer des Algorithmus an.
3) Hinzufügung der Zeit vom Algorithmus.

Nicht-funktionale Anforderungen:

1) Benutzeroberfläche vom NetBenas Fenster.
2) Benutzeroberfläche von Excel.

3.2. Design und Produktion

Den Design von Excel können sie hier im Bild sehen.Die Gestaltung von Excel wird dem Benutzer überlassen.

	A	B	C	D	E
1	Algorithms	First_number	Second_number	gcd	Time
2	The ExtendedEuclidean GCDalgorithm	64227	51287	1	7270.99
3	The Euclidean GCDalgorithm	93338	63707	7	3984.8
4					

Abbildung 1. Excel Fenster von den Daten

Hier unten ist ein Bild von den Leistungen.

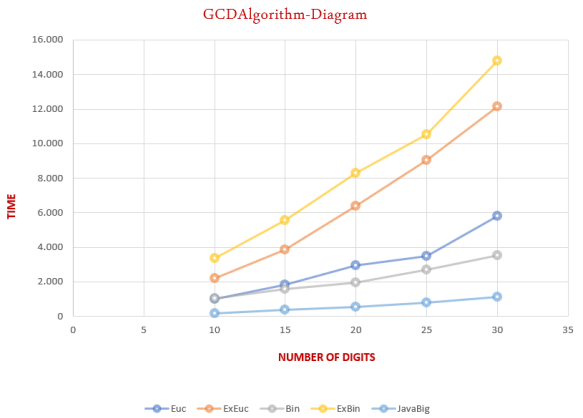

Abbildung 2. Excel Diagramm

Wie man sieht ist der Java BigInteger der schnellste aber für den Algorithmus gibt es keine Erklärung.Deswegen wird der zweit schnellste genommen und das ist der Binärer ggT Algorithmus.Er hat eine hohe Leistung wegen seiner

RightShiftOperation weil das teilt die Zahlen immer durch 2.Er benutzt nicht wie der Euklidische Algorithmus die **modulo operation**.

3.3. Bewertung

Die Anforderungen wurden erfolgreich erfüllt, das kann man an den Bildern erkennen.

4. Ein NetBeans/Java Programm für die Berechnung des ggT von großen Zahlen

4.1. Anforderungen

Funktionale Anforderungen:

1) Berechnung von den Zufallszahlen.
2) Berechnung von dem größten gemeinsamen Teiler.

Nicht-funktionale Anforderungen:

1) Die TextField Funktion.
2) Die Verfügbarkeit von Funktionen.

4.2. Design und Produktion

Über den NetBeans Design gibt es mehr im finalen Bericht zu zeigen, hier ist nur ein Bild.

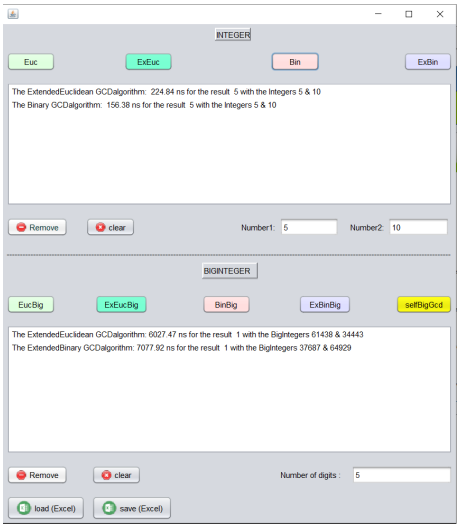

Abbildung 3. NetBeans Fenster

Hier wird nur ein kleiner Teil vom Code gezeigt.

```
int k=0;
while((!(a.testBit(0))) && (!(b.testBit(0))))
{
    a=a.shiftRight(1);
    b=b.shiftRight(1);

    k++;
}
```

Abbildung 4. Ein Teil vom Binären Code

1) Als erstes kontrolliert man, ob die Zahlen nicht 0 sind.
2) Dann schaut man wie oben im Bild mit einer **while Schleife**, ob die Zahlen gerade sind damit man sie durch 2 teilen kann mit der **RightShiftOperation.TestBit(0)** ist eine Funktion von BigIntegr, um binär zu schauen , ob die zahl gerade ist.Es schaut, ob an der ersten binären Stellenzahl eine 1 ist, wen da eine ist,dann ist die Zahl ungerade.
3) Jedes mal wenn es durch 2 teilt, wird der Variabel **k** eine 1 hinzugefügt.
4) (Nicht mehr im Bild) Es wird immer geprüft, ob einer der zwei Zahlen gerade ist bis sie ungerade werden.Zum Schluss wenn die Zahl **a** größer als **b** ist dann werden sie vertauscht und **b** wird von **a** subtrahiert, bis **b** null wird.So gibt der Code ,durch die **LeftShiftOperation (Multiplikation)** von **a** und **k** ,den ggT zurück .

4.3. Bewertung

An den Bildern erkennt man,dass die Anforderungen erfolgreich erfüllt wurden.

5. Abschluss und Anerkennung

Schlussendlich wurde das Ziel erreicht und herausgefunden das der Binärer ggT Algorithmus die zweit höchste Leistung nach dem Java Algorithmus hat .

Ich,Esada Licina danke meinem Tutor Volker Müller,der BICS Gruppe,meiner Familie und Freunden für die Unterstützung.