7. November 2013

HAW Hamburg

Dokumentation

Zu Aufgabenblatt 03 aus der Vorlesungsreihe „Algorithmen und Datenstrukturen“

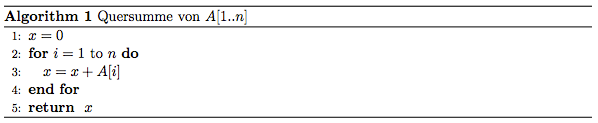
Dokumentation

Zu Aufgabenblatt 03 aus der Vorlesungsreihe „Algorithmen und Datenstrukturen“

# ÜbungsAufgabe 3.1

## TeilAufgabe 1

*Bestimmen Sie Anzahl der Operationen, die der folgende Algorithmus ausführt:*

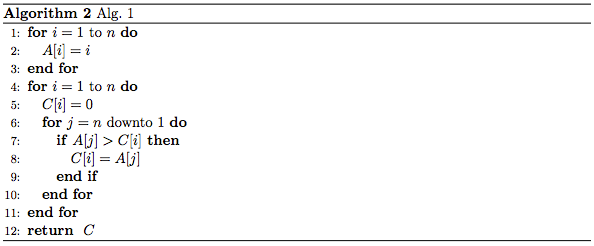
**

In Zeile 1 wird eine Zuweisung gemacht. (1)  
In Zeile 2 wird erst eine Zuweisung gemacht und in jedem darauf folgenden Durchlauf i inkrementiert. (n)  
In Zeile 3 wird eine Addition und eine Zuweisung durchgeführt. Allerdings passiert das in einem Assemblertakt, sodass wir dies nur als eine Operation werten. Trotzdem wird diese Operation n mal durchgeführt. (n)  
In Zeile 5 wird eine Rückgabe gemacht. (1)

Zusammen beträgt die Laufzeit T(2n + 2)

## Teilaufgabe 2

*Bestimmen Sie Anzahl der Operationen, die der folgende Algorithmus ausführt:*

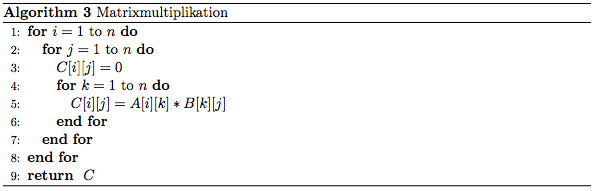
**

In Zeile 1 wird erst eine Zuweisung gemacht und in jedem darauf folgenden Durchlauf i inkrementiert. (n)   
In Zeile 2 wird eine Zuweisung gemacht. Diese findet n mal statt. (n)   
In Zeile 4 wird erst eine Zuweisung gemacht und in jedem darauf folgenden Durchlauf i inkrementiert. (n)  
In Zeile 5 wird eine Zuweisung gemacht. Diese findet n mal statt. (n)  
In Zeile 6 wird erst eine Zuweisung gemacht und in jedem darauf folgenden Durchlauf i inkrementiert. Dies wird n mal gemacht (n²)  
In Zeile 7 wird ein Vergleich durchgeführt. Dies findet n² mal statt. (n²)  
In Zeile 8 findet eine Zuweisung statt. Diese kann maximal n² mal statt finden. (n²)  
In Zeile12 findet eine Rückgabe statt. (1)

Zusammen beträgt die Laufzeit T(3n² + 4n + 1)

## Teilaufgabe 3

*Bestimmen Sie Anzahl der Operationen, die der folgenden Algorithmus ausführt:*

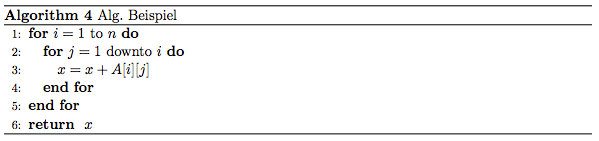


In Zeile 1 wird erst eine Zuweisung gemacht und in jedem darauf folgenden Durchlauf i inkrementiert. (n)   
In Zeile 2 wird erst eine Zuweisung gemacht und in jedem darauf folgenden Durchlauf i inkrementiert. Dies wird n mal gemacht (n²)  
In Zeile 3 findet eine Zuweisung statt. Dieses passiert n² mal. (n²)  
In Zeile 4 wird erst eine Zuweisung gemacht und in jedem darauf folgenden Durchlauf i inkrementiert. Dies wird n² mal gemacht (n³)  
In Zeile 5 findet eine Multiplikation und eine Zuweisung statt. Da dies in einem Assemblertakt passiert, werten wir dies als eine Operation. Das findet n³ mal statt. (n³)  
In Zeile 9 findet eine Rückgabe statt. (1)

Zusammen beträgt die Laufzeit T(2n³ + 2n² + n + 1)

## Teilaufgabe 4

*Bestimmen Sie Anzahl der Operationen, die der folgenden Algorithmus ausführt:*



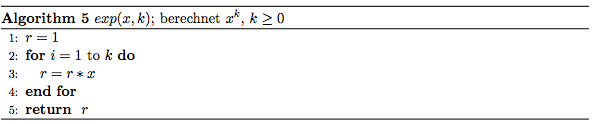
In Zeile 1 wird erst eine Zuweisung gemacht und in jedem darauf folgenden Durchlauf i inkrementiert. (n)   
In Zeile 2 wird erst eine zuweisung gemacht und in jedem darauf folgenden Durchlauf I inkrementiert. Dies findet n mal statt, allerdings veringert sich die Menge der Inkrementationen bei jedem Durchlauf um 1. Daher lässt sich der Aufwand mit der Gaußschen Summenformel beschreiben. ((n² + n) / 2)  
In Zeile 3 findet eine Addition und eine Zuweisung statt. Da dies in einem Prozessortakt statt findet werten wir dies als eine Operation. Diese findet (n² + n) / 2 mal statt. ((n² + n) / 2)   
In Zeile 6 findet eine Rückgabe statt. (1)

Zusammen beträgt die Laufzeit T(n² + 2n + 1)

# übungsaufgabe 2

## Teilaufgabe 1

*Implementieren Sie folgendes Verfahren zum Potenzieren von x :*



Das Verfahren wurde implementiert. Die Implementierung befindet sich im Anhang zu dieser Dokumentation.

## Teilaufgabe 2

*Nutzen Sie folgende Gleichungen aus, um Potenzen schneller zu berechnen:*

## Teilaufgabe 3

*Vergleichen Sie die Laufzeit der Algorithmen in Abhängigkeit von k! Machen Sie geeignete*  
*Experimente und stellen Sie die beiden Laufzeiten graphisch dar.*

*Für welche Werte von x und k kommen die Implementation an ihre Grenzen?*

Die Obere Grenze von expOpt liegt bei k = 30 für n = 2.  
Die Obere Grenze von exp liegt ebenso bei k = 30 für n = 2.  
Die Obere Grenze von expOpt liegt bei n = 46340 für k = 2.  
Die Obere Grenze von exp liegt bei n = 46340 für k = 2.

Daraus ist zu entnehmen, dass die optimierte Version von expOpt keine höheren Zahlen zur Berechnung benötigt als exp und so dieselben Grenzen hat wie exp, dafür aber wesentlich schneller ist.

## Teilaufgabe 4

*Beide Implementation können auch dann eingesetzt werden, wenn x keine Zahl, sondern eine Matrix ist.*   
*Vergleichen Sie die Laufzeiten, indem Sie wieder quadratische (geeignet große) Zufallsmatrizen erzeugen und diese Potenzieren.*

# Übungsaufgabe 3

## teilaufgabe 1

*Beweisen sie:*

Zeigen durch:

Einsetzen:

Da 0 offensichtlich kleiner als Unendlich ist, ist die Aussage wahr.

## Teilaufgabe 2

*Beweisen sie:*

Zeigen durch:

Einsetzen:

Da das Ergebnis nicht < unendlich ist, ist die Aussage wahr.

## teilaufgabe 3

*Betrachte*

### a)

*Geben Sie eine Funktion an, für die und für alle n ab einem n0 gilt.*

Wir wählen

Da der höchste Exponent von f kleiner als der von g ist, ist es offensichtlich, dass

Ab einem n0 von 0 gilt, dass

Wie durch die Konstante +3 in zu sehen ist.

### B)

*Geben Sie eine Funktion an, für die und für alle n ab einem n0 gilt.*

Wir wählen

Da f und g bis auf die Konstanten identisch sind gilt

Ab einem n0 von 0 gilt, dass

Wie durch die größere Konstante +4 in und sonstige Gleichheit von f und g zu sehen ist.

## Teilaufgabe 4

*Wir betrachten Polynome mit natürlichzahligen Koeffizienten, d.h. Funktionen der Form  
 mit und wenn . Das Polynom hat dann den Grad k.*   
*Zeigen Sie: Für zwei Polynome f und g mitgleichem Grad gilt .*

Da keiner der Summanden 0 sein kann ( durch ) ist der einzig relevante Summand   
Daher müssen wir nur zeigen, dass die beiden Summanden mit dem größten Exponenten in der selben Komplexitätsklasse sind.

Weil f und g denselben Grad haben, gilt

Wir wählen und   
Daher gilt:

Und somit ist

## Teilaufgabe 5

*Zeigen Sie : Sei . Es gilt*

Da bei einer Summation von Polynomen mit natürlichzahligen Koeffizienten nur derjenige für die Komplexität der Funktion f relevant ist, der den größten Exponenten hat, können wir sagen, dass

Also reicht es zu zeigen, dass

Was offensichtlich wahr ist.