Aufgabe 01 - Komplexität von Algorithmen

Komplexitätsanalyse von Algorithmen

Für die Komplexitätsanalyse von Algorithmen ist die Laufzeit besonders spannend und jeder Algorithmus kann in Bezug auf seine Laufzeit in verschiedene Laufzeitklassen eingeordnet werden. In diesem Seminar wollen wir uns mit der mathematischen Berechnung der Laufzeit eines Algorithmus, sowie mit der empirischen Messung einer konkreten Implementierung beschäftigen.

Komplexitätsanalyse Sortierungsalgorithmus

Analysieren Sie den unten stehenden Algorithmus zur Sortierung von Zahlenreihen und vollziehen Sie dessen Funktionsweise nach. Geben Sie die Laufzeit des Algorithmus in T(n) an. Beschreiben Sie welche Zeile im Algorithmus wie zu der Funktion T(n) beiträgt. Ermitteln Sie eine Abschätzung in der O-Notation (oder auch in Θ-Notation). In welcher Komplexitätsklasse liegt der Algorithmus?

```
   Function selectionSort(A):
1    for j = 1 to n-1:
2      // find Position of Minimum in A[j],…,A[n]
3      minpos = j
4      for i = j + 1 to n:
5        if A[i] < A[minpos]:
6          minpos = i
7      // swap values if necessary
8      if minpos > j:
9        temp = A[minpos]
10       A[minpos] = A[j]
11       A[j] = temp
```

Abbildung 1: Sortierungsalgorithmus

Komplexitätsanalyse Matrizenberechnungen

Implementieren Sie hierfür ein Programm, welches sich mit Operationen auf Matrizen beschäftigt. Sie müssen zumindest Matrizen erzeugen, addieren und multiplizieren können. Entwerfen Sie einen Algorithmus, der eine Matrizenmultiplikation zweier N × N -Matrizen durchführt. Verwenden Sie für Ihre Implementierung keine Bibliotheken, sondern implementieren Sie die Multiplikation selbst. Testen Sie Ihre Implementierung mit großen Matrizen, um so die Auswirkungen der Problemgröße auf die Laufzeit zu demonstrieren. Implementieren Sie hierfür

einen Generator der eine Matrix einer gegebenen Dimension zufällig erzeugen kann. Verwenden Sie diesen zur Erzeugung mehrerer Matrizen bis zu einer Größenordnung von 10000x10000.

Führen Sie für zwei der von Ihnen generierten Matrizen eine Multiplikation durch und messen Sie die für die Multiplikation benötigte Laufzeit. Führen Sie iterativ Multiplikationen mit immer größeren Matrizen durch. Wie entwickelt sich hier die Laufzeit? Für die Visualisierung der Laufzeiten eignet sich ein Liniendiagramm sehr gut.

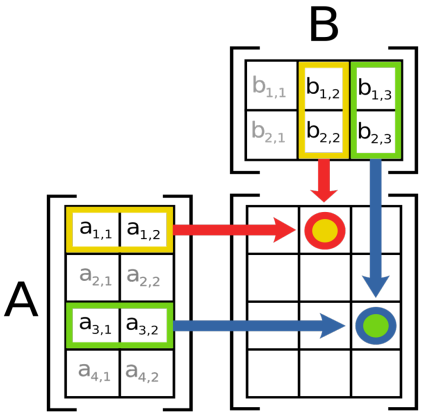

Abbildung 2: By File:Matrix multiplication diagram.svg:User:BilouSee below. - This file was derived from: Matrix multiplication diagram.svg, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=15175268

Ergänzen Sie Ihre Implementierung um eine Matrizenaddition und stellen Sie die Laufzeitwerte denen der Multiplikation gegenüber.

Geben Sie die Laufzeit Ihres Algorithmus in T (n) an. Beschreiben Sie welche Zeile in Ihrem Algorithmus wie zu der Funktion T (n) beiträgt. Ermitteln Sie eine Abschätzung in der O-Notation (oder auch in Θ-Notation). In welcher Komplexitätsklasse liegt Ihr Algorithmus? Geben Sie die Laufzeit sowohl für die Multiplikation als auch die Addition an. Zeigen Sie, dass die angenommene Komplexitätsklasse tatsächlich korrekt ist und finden Sie Werte für die Konstanten c1, c2 und n0.

Die Implementierung kann in einer beliebigen Sprache erfolgen.