HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG, DEPARTMENT INFORMATIK

Algorithmen und Datenstrukturen - Aufgabe 4

Hauke Goldhammer, Markus Blechschmidt

18. April 2017

ABSTRACT

In dieser Aufgabe soll sich mit rekursiven und iterativen Verfahren am Beispiel des Pascalschen Dreiecks beschäftigt werden. Dieses Dokument stellt einen Lösungsvorschlag dar und beinhaltet die Aufbereitung und heuristische Auswertung der Schrittzähler in den Implementationen, so wie eine analytische Herleitung.

INHALTSANGABE

ABBILDUNGEN

1 Algorithmen

Dieser Abschnitt beschreibt grundlegend die Funktion und Implementation der Algorithmen. Alle Algorithmen implementieren das gleiche Interface. Dieses beschreibt, dass die Funktionen als einzigen Parameter entgegennimmt, welche Zeile errechnet werden soll. Der Rückgabetyp ist ein Zahlenfeld, welches die entsprechenden Zeile darstellt.

1.1 Rekursiv

Die Abbruchbedienung für diese Implementation ist, dass man die 1. oder eine “kleine” Zeile errechnen lässt. In diesem Fall wird $[1,1]$ zurückgegeben. In alles anderen Fällen wird erst die vorhergehende Zeile errechnet und dann damit iterativ die Elemente der nächste Zeile generiert.

1.2 Iterativ

Dieser Algorithmus generiert als erstes die erste Zeile. Danach werden nach und nach darauf basierend alle Weiteren Zeilen aus der vorhergehenden generiert, bis man die gewünschte Zeile erreicht hat.

1.3 Direkt

Dieser Algorithmus generiert alle Elemente der Zeile mit dem Binominialkoeffizienten $\binom{n}{k}$, wobei n die Zeilennummer ist und k die Position in der Zeile.

2 Laufzeitergebnisse

Die Abbildung 2.1 zeigt grafisch den Zusammenhang zwischen Problemgröße N, welche die Nummer der Reihe des Pascalschen Dreiecks darstellt, welche errechnet werden soll, und $T(N)$, welches das Ergebnis unseres eingebauten Schrittzählers ist.
Bereits an Abbildung 2.1 ist zu erkennen, dass der iterative und rekursive Algorithmus ein stärkeres Aufwandswachstum aufweisen als der direkte Algorithmus. Dies zeigt sich auch noch mal deutlich in der Abbildung 2.2, in welcher beide Achsen des Koordinatensystems gleich skaliert sind. Hieran kann man über die Steigung auch ablesen, dass der interative und rekursive Algorithmus ein quadratisches Wachstum aufweisen (Steigung 2) und der direkte Algorithmus ein lineares (Steigung 1).

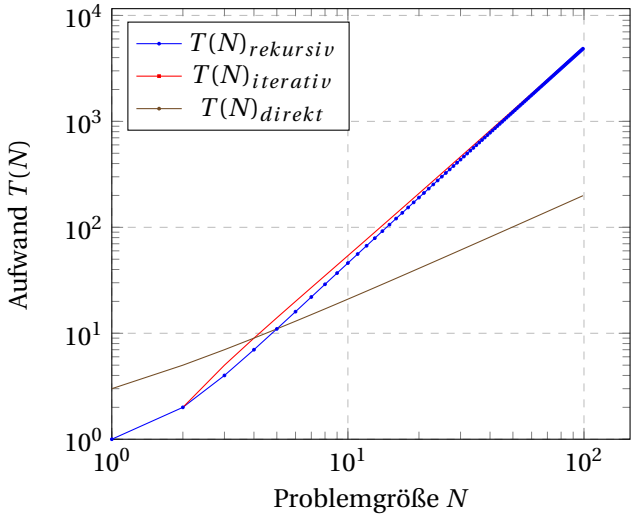

Abb. 2.1: Graph des Aufwands $T(N)$ für alle drei Funktionen bei steigender Aufwandsgröße N in einer log-log-Skala.

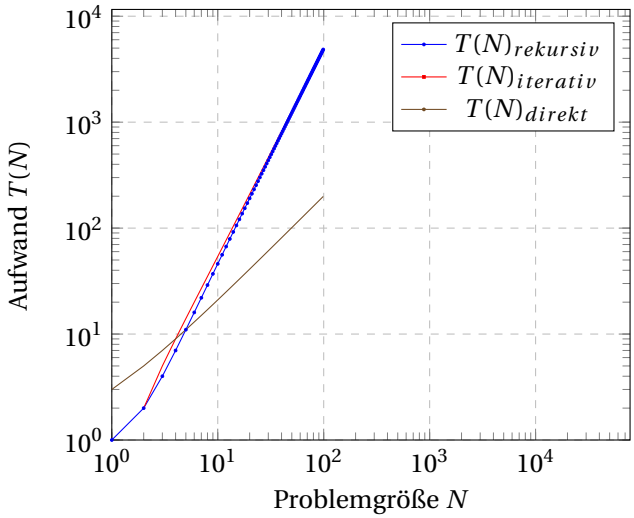

Abb. 2.2: Graph des Aufwands $T(N)$ für alle drei Funktionen bei steigender Aufwandsgröße N in einer symmetrischen log-log-Skala.

Die quadratischen Laufzeiten der iterativen und rekursiven Lösung lassen sich mit der Gaußschen Summenformel erklären. Für jede Zeile braucht man die Zeile davor und jede Zeile braucht einen Rechenschritt mehr als die davor.

$$T(N)_{rekursiv} = \begin{cases} 0 & ,N \leq 1 \\ T(N-1)_{rekursiv} + N - 1 & ,N > 1 \end{cases}$$

somit gilt

$$T(N)_{rekursiv} = \mathcal{O}(n^2) \tag{2.1}$$

$$T(N)_{iterativ} = \sum_{i=0}^{N-1} i$$

somit gilt

$$T(N)_{rekursiv} = \mathcal{O}(n^2) \tag{2.2}$$

$$T(N)_{direkt} = \left(\sum_{i=0}^{N-1} 1\right) + \left(\sum_{i=0}^{N-1} 1\right) = \sum_{i=0}^{N-1} 2$$

somit gilt

$$T(N)_{direkt} = \mathcal{O}(n) \tag{2.3}$$