Universität Konstanz
Fachbereich Informatik & Informationswissenschaft
Prof. Dr. Sven Kosub / Michael Aichem, Julian Müller, Dagmar Sorg, Michael Strecke, Nadja Willenborg

Mathematik: Diskrete Strukturen
SS 2015

6. Übungsblatt
(Musterlösung)

Ausgabe: 22.05.2015 **Abgabe:** 29.05.2015, bis spätestens 12:00 per Mail an den Tutor

Vertiefung: **10 Punkte**

(a) Welche Potenzreihe (in möglichst einfacher Darstellung mit nur einem Summensymbol) entspricht dem Produkt der Potenzreihen $\sum_{n=0}^{\infty} x^n$ und $\sum_{n=0}^{\infty}(2n+1)\cdot x^n$?

(b) Es sei $A(x)$ die erzeugende Funktion der Folge $a_0, a_1, \ldots$. Welche Folge besitzt dann die erzeugende Funktion $\int_0^x A(t)dt$?

(c) Welche erzeugende Funktion hat die Folge $(1,1,1,1,1,1,0,0,\ldots,0,\ldots)$?

(d) Welche erzeugende Funktion hat die Folge $(0,0,0,0,0,0,1,1,\ldots,1,\ldots)$?

(e) Welche erzeugende Funktion hat die Folge $(3,2,1,3,2,1,3,2,1,\ldots,3,2,1,\ldots)$?

(f) Welche erzeugende Funktion hat die Folge $(-1,1,-1,1,-1,1,\ldots,-1,1,\ldots)$?

(g) Welche erzeugende Funktion hat die Folge $(0,1,3,6,10,15,21,28,36,\ldots,?,\ldots)$?

(h) Ist $a_n =_{\text{def}} (1+\sqrt{2})^n + (1-\sqrt{2})^n$ für alle $n \in \mathbb{N}$ stets eine natürliche Zahl?

Hinweis: Finden Sie eine geeignete rekursive Darstellung für a_n.

(i) Gegeben sei folgende lineare Rekursionsgleichung (aber ohne fester Ordnung!):

$$\begin{aligned} a_n &=_{\text{def}} 1 + \sum_{k=0}^{n-1} a_k \qquad \text{für } n \geq 1 \\ a_0 &=_{\text{def}} 1 \end{aligned}$$

Bestimmen Sie die Folgenglieder a_n mittels der erzeugenden Funktion.

(j) Bestimmen Sie die Folgenglieder zur Folge aus Teilaufgabe (i), indem Sie $a_n - a_{n-1}$ betrachten und eine einfacherere Rekursionsgleichung gewinnen.

Lösung:

(a)

$$\sum_{n=0}^{\infty} x^n \cdot \sum_{n=0}^{\infty}(2n+1)\cdot x^n = \sum_{k=0}^{\infty}\left(\sum_{n=0}^{k}(2n+1)\right)x^k = \sum_{k=0}^{\infty}(k(k+1)+(k+1))x^k$$
$$= \sum_{k=0}^{\infty}(k^2+2k+1)x^k = \sum_{k=0}^{\infty}(k+1)^2x^k$$

(b) Aus $\int a_n x^n \mathrm{d}x = \frac{a_n}{n+1}x^{n+1}$ folgt, dass $\int_0^x A(t)\mathrm{d}t$ die Folge $0, a_0, \frac{1}{2}a_1, \frac{1}{3}a_2, \ldots$ besitzt.

(c) Es gilt

$$A(x) = \sum_{n=0}^{5} x^n = \sum_{n=0}^{\infty} x^n - x^6 \sum_{n=0}^{\infty} x^n = (1-x^6)\cdot\sum_{n=0}^{\infty} x^n = \frac{1-x^6}{1-x}.$$

(d) Mit (c) findet man

$$A_{1,1,1,\ldots}(x) - A_{(c)}(x) = \frac{1}{1-x} - \frac{1-x^6}{1-x} = \frac{x^6}{1-x}.$$

(e)

$$F(x) = \sum_{n=0}^{\infty} F_n x^n = 3\sum_{n=0}^{\infty} x^{3n} + 2\sum_{n=0}^{\infty} x^{3n+1} + 1\sum_{n=0}^{\infty} x^{3n+2} = (3+2x+x^2)\sum_{n=0}^{\infty} x^{3n}$$
$$= \frac{3+2x+x^2}{1-x^3}.$$

(f) Wir trennen die Summe nach Summanden mit geradem und ungeradem Index:

$$A(x) = \sum_{n=0}^{\infty} a_n x^n = \sum_{n=0}^{\infty} a_{2n+1}x^{2n+1} + \sum_{n=0}^{\infty} a_{2n}x^{2n} = \sum_{n=0}^{\infty} x^{2n+1} - \sum_{n=0}^{\infty} x^{2n}$$
$$= x\sum_{n=0}^{\infty} x^{2n} - \sum_{n=0}^{\infty} x^{2n} = (x-1)\sum_{n=0}^{\infty} x^{2n} = \frac{x-1}{1-x^2} = \frac{-1}{1+x}$$

Alternative Lösung 1:

$$A(x) = \sum_{n=0}^{\infty} -(-x)^n = -1 + \sum_{n=1}^{\infty} -(-x)^n = -1 + \sum_{n=0}^{\infty} -(-x)^{n+1}$$
$$= -1 - x\sum_{n=0}^{\infty} -(-x)^n = -1 - xA(x) = \frac{-1}{1+x}$$

Alternative Lösung 2: Substituiere $y := -x$. Dann gilt $A(y) = -\sum_{n=0}^{\infty} y^n = \frac{-1}{1-y}$, und Rücksubstitution ergibt $A(x) = \frac{-1}{1+x}$.

(g) Mit der Tabelle (Skript S. 30) und der Beobachtung, dass die Folge durch den Ausdruck $a_n = \frac{n(n+1)}{2}$ gegeben ist, findet man:

$$\begin{aligned} A(x) &= \sum_{n=0}^{\infty} \frac{n(n+1)}{2} x^n = \frac{1}{2} \sum_{n=0}^{\infty} n^2 x^n + \frac{1}{2} \sum_{n=0}^{\infty} n x^n = \frac{1}{2} \left(\frac{x(1+x)}{(1-x)^3} + \frac{x}{(1-x)^2} \right) \\ &= \frac{x}{(1-x)^3} \end{aligned}$$

(h) Vgl. Theorem 2.5:
$A = 1$, $B = -1$, $\alpha = 1 + \sqrt{2}$, $\beta = 1 - \sqrt{2}$.
α und β sind Lösungen von $t^2 - a_1 t - a_2 = 0$. Dies ergibt $a_1 = 2$ und $a_2 = 1$. Somit folgt $a_n = 2 \cdot a_{n-1} + a_{n-2}$ und a_n ist stets eine natürliche Zahl.

(i) Mittels Rückwärts-Konvolution formen wir geeignet nach $A(x)$ um:

$$\begin{aligned} A(x) &= \sum_{n=0}^{\infty} a_n x^n = 1 + \sum_{n=1}^{\infty} a_n x^n = 1 + \sum_{n=1}^{\infty} \left(1 + \sum_{k=0}^{n-1} a_k \right) x^n = 1 + x \sum_{n=0}^{\infty} \left(1 + \sum_{k=0}^{n} a_k \right) x^n \\ &= 1 + x \sum_{n=0}^{\infty} x^n + x \cdot \left(\sum_{n=0}^{\infty} a_n x^n \right) \left(\sum_{n=0}^{\infty} x^n \right) = 1 + \frac{x}{1-x} + \frac{x}{1-x} \cdot A(x) \\ &= \frac{1}{1-2x} \end{aligned}$$

Damit haben wir in der üblichen Darstellung (vgl. Skript) $A = 1, B = 0, \alpha = 2, \beta = 0$ und somit

$$A_n = 2^n.$$

(j) Wegen

$$a_n - a_{n-1} = 1 + \sum_{k=0}^{n-1} a_k - \left(1 + \sum_{k=0}^{n-2} a_k \right) = a_{n-1} \quad \Longleftrightarrow \quad a_n = 2a_{n-1}$$

gilt offensichtlich $a_n = 2^n$.

Kreativität: **10 Punkte**

Betrachten Sie die allgemeine inhomogene Version der FIBONACCI-Zahlen:

$$\begin{aligned} F_n &=_{\text{def}} F_{n-1} + F_{n-2} + g(n) \qquad \text{für } n \geq 2 \\ F_1 &=_{\text{def}} g(1) \\ F_0 &=_{\text{def}} g(0) \end{aligned}$$

Hierbei ist $g : \mathbb{N} \to \mathbb{R}$ eine beliebige Funktion.

(a) Bestimmen Sie die erzeugende Funktion unter Verwendung von $G(x) =_{\text{def}} \sum_{n=0}^{\infty} g(n) \cdot x^n$.

(b) Lösen Sie die Rekursionsgleichung für die Funktion $g(n) =_{\text{def}} 1$.

Lösung:

(a) Wir bestimmen die erzeugende Funktion:

$$\begin{aligned} F(x) &:= \sum_{n=0}^{\infty} F_n x^n \\ &= g(0) + g(1)x + \sum_{n=2}^{\infty} F_n x^n \\ &= g(0) + g(1)x + \sum_{n=2}^{\infty} (F_{n-1} + F_{n-2} + g(n)) x^n \\ &= g(0) + g(1)x + \sum_{n=2}^{\infty} g(n)x^n + \sum_{n=2}^{\infty} F_{n-1}x^n + \sum_{n=2}^{\infty} F_{n-2}x^n \\ &= \sum_{n=0}^{\infty} g(n)x^n + x \sum_{n=1}^{\infty} F_n x^n + x^2 \sum_{n=0}^{\infty} F_n x^n \\ &= G(x) - g(0)x + xF(x) + x^2 F(x) \end{aligned}$$

Mithin folgt $F(x) = \dfrac{G(x) - g(0)x}{1 - x - x^2}$.

(b) Aus $g(n) = 1$ folgt $G(x) = \dfrac{1}{1-x}$ und $F(x) = \dfrac{1 - x(1-x)}{(1-x)(1-x-x^2)} = \dfrac{1 - x + x^2}{(1-x)(1-x-x^2)}$.

Wir ermitteln die zugehörige Potenzreihe mittels Partialbruchzerlegung. Wir suchen A, B, C, α, β, sodass

$$\frac{A}{1-\alpha x} + \frac{B}{1-\beta x} + \frac{C}{1-x} = \frac{1 - x + x^2}{(1-x)(1-x-x^2)}$$

Aus der entsprechenden Rechnung im Skript zur Fibonacci-Reihe wissen wir, dass $\alpha = \frac{1+\sqrt{5}}{2}$ und $\beta = \frac{1-\sqrt{5}}{2}$ sind.

Für den Zähler erhalten wir:

$$\begin{aligned}1 - x + x^2 &= A(1-\beta x)(1-x) + B(1-\alpha x)(1-x) + C(1-\alpha x)(1-\beta x)\\ &= A - A(\beta+1)x + A\beta x^2 + B - B(\alpha+1)x + B\alpha x^2 + C - C(\alpha+\beta)x + C\alpha\beta x^2\\ &= (A\beta + B\alpha + C\alpha\beta)x^2 - (A(\beta+1) + B(\alpha+1) + C)x + (A+B+C)\end{aligned}$$

Aus dem Koeffizientenvergleich ergeben sich die Gleichungen:

$$A + B + C = 1 \tag{1}$$
$$A(\beta+1) + B(\alpha+1) + C = 1 \tag{2}$$
$$A\beta + B\alpha + C\alpha\beta = 1 \tag{3}$$

Die Lösung dieses Gleichungssystems ist $A = \frac{1+\sqrt{5}}{\sqrt{5}}$, $B = \frac{4}{\sqrt{5}(1+\sqrt{5})}$ and $C = -1$. Daher gilt:

$$\begin{aligned}\sum_{n=0}^{\infty} F_n x^n &= F(x)\\ &= \frac{1-x+x^2}{1-x-x^2}\\ &= \frac{A}{1-\alpha x} + \frac{B}{1-\beta x} + \frac{C}{1-x}\\ &= A\sum_{n=0}^{\infty}(\alpha^n x^n) + B\sum_{n=0}^{\infty}(\beta^n x^n) + C\sum_{n=0}^{\infty} x^n\end{aligned}$$

Koeffizientenvergleich ergibt:

$$F_n = A\alpha^n + B\beta^n + C = \frac{1+\sqrt{5}}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n + \frac{4}{\sqrt{5}(1+\sqrt{5})}\left(\frac{1-\sqrt{5}}{2}\right)^n - 1$$

Transfer: **10 Punkte**

In der Projekt-Dokumentation zu einem *Software Code Review* sind Sie auf die Pseudocode-Beschreibung eines implementierten Sortierverfahrens gestoßen, zusammen mit der Information, dass zum Sortieren von n Integer-Zahlen, die in einem Array A gespeichert sind, der Hauptaufruf mit SortArray$(A, 0, n-1)$ erfolgen muss:

Algorithmus:	ArraySort(A, i, j)
Eingabe:	Integer-Array A der Länge n, Indizes $i, j \in \{0, n-1\}$, $i \leq j$
Ausgabe:	–

```
1.   IF i = j                                  /* Terminierungsfall, Array der Länge 1
2.     RETURN
3.   ELSE
4.     IF i + 1 = j                            /* Terminierungsfall, Array der Länge 2
5.       IF A[i] > A[j]                                  /* Inhalte falsch angeordnet
6.         Vertausche Inhalte von A[i] und A[j]
7.       RETURN
8.     ELSE
9.       k ← ⌊(j − i + 1)/3⌋          /* Drittel der Länge berechnen, abrunden
10.        ArraySort(A, i, j − k)                 /* vordere zwei Drittel sortieren
11.        ArraySort(A, i + k, j)                 /* hintere zwei Drittel sortieren
12.        ArraySort(A, i, j − k)    /* vordere zwei Drittel nochmal sortieren
13.  RETURN
```

Sie wollen sich von der Korrektheit und der Effizienz des Verfahrens überzeugen.

(a) Zeigen Sie, dass der Sortieralgorithmus mit Aufruf ArraySort$(A, 0, n-1)$ ein Array A der Länge n aufsteigend sortiert.

Verwenden Sie dazu vollständige Induktion über die Länge der (Teil)Arrays, d.h., beweisen Sie, dass aus der Korrektheit des Algorithmus für alle (Teil)Arrays der Länge $m \leq n-1$ die Korrektheit des Algorithmus für alle (Teil)Arrays der Länge n folgt.

Hinweis: Überlegen Sie sich, dass nach dem zweiten rekursiven Aufruf von SortArray die k größten Zahlen bereits an der richtigen Position im Array stehen.

(b) Die Laufzeit des Algorithmus hängt maßgeblich von der Anzahl der Vergleiche „$A[i] > A[j]$“ in Zeile (5) ab (aber nicht von den im Array gespeicherten Integer-Zahlen!). Geben Sie eine Rekursionsgleichung für die Anzahl dieser Vergleiche V_n auf einem Array der Länge n.

Wie schnell läuft der Algorithmus in Abhängigkeit von n?

Lösung:

(a) Wir zeigen die Behauptung mit vollständiger Induktion über die Länge n des zu sortierenden Arrays.
Induktionsanfang für $n = 1$ bzw. $n = 2$: Wird vom Algorithmus in den Zeilen 1-2 bzw. 4-7 korrekt gelöst.
Induktionsschritt: Wir nehmen an, dass der Algorithmus alle Arrays der Länge $m \leq n-1$

korrekt sortiert. Betrachte im Folgenden den Aufruf ArraySort$(A, 0, n-1)$, d.h. wir möchten ein Array der Länge n sortieren. Der Algorithmus sortiert in Zeile 10 die vorderen zwei Drittel des Arrays korrekt (nach Induktionsvoraussetzung, da $\lceil \frac{2}{3} \cdot n \rceil < n$ für $n \geq 3$). Ebenso sortiert der Algorithmus in Zeile 11 die hinteren zwei Drittel korrekt (wieder nach Induktionsvoraussetzung).

Zwischenbeobachtung: Nachdem die vorderen zwei Drittel sortiert wurden, steht keine der k größten Zahlen im ersten Drittel des Arrays. Nun werden die hinteren zwei Drittel sortiert. Danach stehen die k größten Zahlen im hinteren Drittel des Arrays.

Somit sind, bevor der Aufruf in Zeile 12 erfolgt, alle Einträge im letzten Drittel des Arrays größer oder gleich den Einträgen in den ersten zwei Dritteln. Der Algorithmus sortiert nun die ersten zwei Drittel und somit ist das ganze Arrays korrekt sortiert.

(b) Die Anzahl der Vergleiche V_n ist:

i. $V_1 = 0$

ii. $V_2 = 1$

iii. $V_n = 3 \cdot V_{\lceil \frac{2n}{3} \rceil}$

Ist n von der Form $n = 2 \cdot (\frac{3}{2})^k$ (hiermit werden Rundungsprobleme vermieden), so löst sich die Rekursion auf zu:

$$\begin{aligned}
V_n &= V_{2\cdot(\frac{3}{2})^k} \\
&= 3 \cdot V_{2\cdot(\frac{3}{2})^{k-1}} \\
&= 3^2 \cdot V_{2\cdot(\frac{3}{2})^{k-2}} \\
&\vdots \\
&= 3^i \cdot V_{2\cdot(\frac{3}{2})^{k-i}} \\
&\overset{i=k}{=} 3^k \cdot V_2 \\
&\overset{V_2=1}{=} 3^k \\
&= 3^{\log_{1,5}(\frac{n}{2})} \\
&= \left(\frac{n}{2}\right)^{\log_{1,5} 3}
\end{aligned}$$

Der Algorithmus zerlegt ein Problem der Größe n in drei Teilprobleme der Größe $\lceil \frac{2n}{3} \rceil$. Die Rekursionsgleichung für die Laufzeit des Algorithmus lautet also:

$$T(n) = 3 \cdot T\left(\left\lceil \frac{2n}{3} \right\rceil\right) + c.$$

Da die Laufzeit des Algorithmus massgeblich abhängig ist von der Anzahl der Vergleiche, ergibt sich eine Laufzeit von: $T(n) \in \Theta(n^{\log_{1,5} 3}) \approx \Theta(n^{2,71}) \subsetneq \Omega(n^2)$.

Die Laufzeit dieses Algorithmus ist somit schlechter als andere Ihnen bekannte Sortieralgorithmen, wie z.B. SelectionSort, QuickSort, MergeSort, HeapSort, BubbleSort etc. Dieser Algorithmus heißt StoogeSort.