Methoden zum Lösen von Rekursionsgleichungen

Bisher wurde „Expandieren der Rekursion“ + „Raten der Gesetzmäßigkeit“ benutzt, um einfache Rekursionsgleichungen zu lösen. Zum Beispiel:

1. Rekursionsgleichung

$$B_1 = 1 \text{ und } B_n = 1 + B_{n/2}$$

   hat die Lösung

$$B_n = 1 + \log n \ .$$

2. Rekursionsgleichung

$$C_1 = 0 \text{ und } C_n = 2C_{n/2} + n$$

   hat die Lösung

$$C_n = n \log n \ .$$

Ziel dieses Abschnittes ist es, weitere Techniken zum Lösen von Rekursionsgleichungen zur Verfügung zu stellen.

Lineare Rekursionsgleichungen k-ter Ordnung

Eine Rekursionsgleichung der Form

$$x_n = a_1 x_{n-1} + \cdots + a_k x_{n-k} + b_k$$

mit den Anfangsbedingungen

$$x_0 = b_0, \ldots, x_{k-1} = b_{k-1}$$

heißt *lineare Rekursionsgleichung k-ter Ordnung*. Wir nennen die Rekursionsgleichung im Falle $b_k = 0$ *homogen* und im Falle $b_k \neq 0$ *inhomogen*.

Homogene Lineare Rekursionsgleichungen 1-ter Ordnung

Expandieren der Rekursion

$$x_n = ax_{n-1} \text{ und } x_0 = b_0$$

liefert die folgende Lösung:

$$x_n = ax_{n-1} = a^2 x_{n-2} = \cdots = a^n x_0 = a^n b_0 \quad .$$

Beispiel: Die Rekursion zur „Weizenkornlegende“ lautet

$$x_n = 2x_{n-1} \ , \ x_0 = 1 \quad .$$

Sie hat die Lösung $x_n = 2^n$. Gemäß der Weizenkornlegende befinden sich auf dem Feld i des Schachbrettes (mit den Feldern $0, 1, \ldots, 63$) demnach 2^i Weizenkörner. Insgesamt also:

$$\sum_{i=0}^{63} 2^i = 2^{64} - 1 = 18.446.744.073.709.551.615$$

Inhomogene Lineare Rekursionsgleichungen 1-ter Ordnung

Expandieren der Rekursion

$$x_n = ax_{n-1} + b_1 \text{ und } x_0 = b_0$$

liefert die folgende Lösung:

$$\begin{aligned}
x_n &= ax_{n-1} + b_1 \\
&= a(ax_{n-2} + b_1) + b_1 = a^2 x_{n-2} + (a+1)b_1 \\
&= a^2(ax_{n-3} + b_1) + (a+1)b_1 = a^3 x_{n-3} + (a^2 + a + 1)b_1 \\
&\dots \\
&= a^n x_0 + (a^{n-1} + a^{n-2} + \cdots + a + 1)b_1 \\
&= \begin{cases} b_0 + nb_1, & \text{falls } a = 1 \\ a^n b_0 + \frac{a^n - 1}{a-1} \cdot b_1, & \text{falls } a \neq 1 \end{cases}
\end{aligned}$$

Beispiel: Sparen mit Zins und Zinseszins

Am Anfang eines Monats: 250 EUR auf's Bankkonto.
Am Ende eines Monats: 0,5% Verzinsung.
Frage: Nach wieviel Jahren ist eine Million angespart?

Für den nach n Monaten angesparten Betrag x_n gilt $x_0 = 0$ und

$$x_n = \left(1 + \frac{0.5}{100}\right)(x_{n-1} + 250) = 1.005x_{n-1} + 251.25 \ .$$

Diese inhomogene lineare Rekursionsgleichung 1-ter Ordnung hat die Lösung

$$x_n = 251.25 \cdot \frac{1.005^n - 1}{0.005} = 50250 \cdot (1.005^n - 1) \ .$$

Dieser Betrag überschreitet 1 Million erst ab $n = 610$ Monaten, also ab 50 Jahren und 10 Monaten.

Homogene lineare Rekursionsgleichungen 2-ter Ordnung

In der Rekursion

$$x_n = a_1 x_{n-1} + a_2 x_{n-2} \text{ und } x_1 = b_1, x_0 = b_0$$

seien a_1, a_2 nicht beide gleich Null. Es seien α, β mit $\alpha \geq \beta$ die Lösungen der Gleichung $t^2 - a_1 t - a_2 = 0$ und

$$A = \begin{cases} \frac{b_1 - \beta b_0}{\alpha - \beta}, & \text{falls } \alpha \neq \beta \\ \frac{b_1 - \alpha b_0}{\alpha}, & \text{falls } \alpha = \beta \end{cases} \quad , \quad B = \begin{cases} \frac{b_1 - \alpha b_0}{\alpha - \beta}, & \text{falls } \alpha \neq \beta \\ b_0, & \text{falls } \alpha = \beta \end{cases} \quad .$$

Dann gilt:

$$x_n = \begin{cases} A\alpha^n - B\beta^n, & \text{falls } \alpha \neq \beta \\ (An + B)\alpha^n, & \text{falls } \alpha = \beta \end{cases} \quad .$$

Im Spezialfall $b_0 = 0, \alpha \neq \beta$ ergibt sich:

$$A = B = \frac{b_1}{\alpha - \beta} \ , \ x_n = \frac{b_1}{\alpha - \beta}(\alpha^n - \beta^n) \ .$$

Anwendung: Fibonacci-Zahlen

Die *Fibonacci-Folge* ist gegeben durch die folgende homogene lineare Rekursionsgleichung 2-ter Ordnung:

$$F_n = F_{n-1} + F_{n-2} \text{ und } F_1 = 1, F_0 = 0 \ .$$

Die Gleichung

$$t^2 - t - 1 = 0 \Leftrightarrow t^2 - t + \frac{1}{4} = \left(t - \frac{1}{2}\right)^2 = \frac{5}{4} \Leftrightarrow t = \frac{1}{2} \pm \frac{\sqrt{5}}{2}$$

hat die Lösungen $\alpha = (1+\sqrt{5})/2$ und $\beta = (1-\sqrt{5})/2$. Es folgt

$$A = B = \frac{1}{\alpha - \beta} = \frac{1}{\sqrt{5}}$$

und somit

$$F_n = \frac{1}{\sqrt{5}} \left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}} \left(\frac{1-\sqrt{5}}{2}\right)^n \ .$$

Beispiel: Kombinatorik auf Wörtern

Für die Anzahl x_n der Wörter aus $\{a, b\}^n$ ohne zwei aufeinander folgende a's gilt nach der Summenregel

$$x_n = x_{n-1} + x_{n-2} \text{ und } x_1 = 2, x_0 = 1 \ .$$

Hierbei repräsentiert x_{n-1} die auf b und x_{n-2} die auf ba endenden Wörter der gewünschten Form. Vergleich mit F_n liefert die Lösung $x_n = F_{n+2}$.

Rekursionsgleichungen vom Typ „Divide&Conquer“

Folgende Rekursionsgleichung modelliert den Zeitaufwand einer rekursiven Prozedur bei Aufteilung eines Problems der Größe n in a Teilprobleme der Größe n/c:

$$T(n) = aT(n/c) + bn \text{ und } T(1) = b \ .$$

Term bn repräsentiert die Anzahl der Rechenschritte zum Aufteilen des Problems in Teilprobleme und zum Zusammenfügen der Teillösungen zu einer Gesamtlösung (Annahme eines „linearen Overhead“). Term $aT(n/c)$ repräsentiert die Anzahl der Rechenschritte zum Lösen der Teilprobleme.

Die Rekursionsgleichung hat die Lösung

$$T(n) = \begin{cases} \theta(n), & \text{falls } a < c, \\ \theta(n \log n), & \text{falls } a = c, \\ \theta(n^{\log_c a}), & \text{falls } a > c. \end{cases}$$

Beweis durch Expandieren der Rekursion

Wir setzen der Einfachheit halber $n = c^k$ voraus. Mit $r = a/c$ ergibt sich:

$$
\begin{aligned}
T(n) &= aT(n/c) + bn \\
&= a(aT(n/c^2) + bn/c) + bn = a^2T(n/c^2) + (r+1)bn \\
&= a^2(aT(n/c^3) + bn/c^2) + (r+1)bn = a^3T(n/c^3) + (r^2+r+1)bn \\
&\cdots \\
&= a^kT_{n/c^k} + (r^{k-1} + \cdots + r + 1)bn = bn \cdot \sum_{i=0}^{k} r^i \ ,
\end{aligned}
$$

wobei $a^kT_{n/c^k} = a^kT_1 = a^kb = (a/c)^kbn = bnr^k$ ausgenutzt wurde. Für $a = c$ ergibt sich wegen $r = 1$ die Lösung $(k+1)bn = \theta(n \log n)$. Für $a < c$ ergibt sich wegen $r < 1$ die Lösung $T(n) = \theta(n)$, da $\sum_{i \geq 0} r^i = 1/(1-r)$ (geometrische Reihe). Für $a > c$ ergibt sich die Lösung

$$T(n) = bn\frac{r^{k+1}-1}{r-1} = \theta(nr^k) = \theta(a^k) = \theta(a^{\log_c n}) = \theta(n^{\log_c a}) \ .$$

Anwendungsbeispiele

Die Laufzeit der rekursiven Prozedur MERGESORT läßt sich abschätzen durch die Rekursionsgleichung

$$T(n) = 2T(n/2) + bn \text{ und } T(1) = b \ .$$

Somit gilt $T(n) = \theta(n \log n)$.

Bei der Multiplikation großer Zahlen benötigten wir beim naiven Verfahren 4 Aufrufe auf Zahlen der halben Bitlänge, wohingegen ein raffinierteres Verfahren nur 3 solche Aufrufe benötigte. Dies führte im ersten Fall auf die Rekursion $T_1(n) = 4T_1(n/2) + bn$ mit Lösung $T_1(n) = \theta(n^{\log 4}) = \theta(n^2)$ und im zweiten Fall auf die Rekursion $T_2(n) = 3T(n/2) + bn$ mit Lösung $T_2(n) = \theta(n^{\log 3}) = \theta(n^{1.58\ldots})$.