**Aufgabe 1:** Rekursive Multiplikation (2+3+3+1 Punkte)

Wir betrachten einen rekursiven Algorithmus zur Multiplikation zweier natürlicher Zahlen $a$ und $b$, welche durch die Ziffernfolgen $a = a_{2n} \dots a_1$ und $b = b_{2n} \dots b_1$ zur Basis $k$ dargestellt seien. Der Algorithmus teilt die Zahlen in jeweils zwei Hälften

$$a_h = a_{2n} \dots a_{n+1} \quad \text{und} \quad a_l = a_n \dots a_1 \quad \text{sowie}$$
$$b_h = b_{2n} \dots b_{n+1} \quad \text{und} \quad b_l = b_n \dots b_1,$$

sodass sie sich als $a = a_h \cdot k^n + a_l$ und $b = b_h \cdot k^n + b_l$ darstellen lassen.

Das Ausmultiplizieren führt auf

$$a \cdot b = (a_h \cdot k^n + a_l)(b_h \cdot k^n + b_l) = a_h b_h k^{2n} + (a_h b_l + a_l b_h) k^n + a_l b_l.$$

Durch eine äquivalente Umformung erhält man

$$a_h b_l + a_l b_h = (a_h + a_l)(b_h + b_l) - a_h b_h - a_l b_l,$$

was in die obige Gleichung eingesetzt schließlich auf

$$a \cdot b = a_h b_h k^{2n} + ((a_h + a_l)(b_h + b_l) - a_h b_h - a_l b_l) k^n + a_l b_l$$

führt. Da die Multiplikation mit Potenzen zur Basis $k$ effizient durch einfache Verschiebungen realisiert werden kann, müssen hier im Wesentlichen nur noch jeweils drei Produkte

$$T_1 = a_h b_h, \qquad T_2 = a_l b_l, \qquad T_3 = (a_h + a_l)(b_h + b_l)$$

rekursiv berechnet werden, verknüpft mittels Verschiebe- und Additionsoperationen:

$$a \cdot b = T_1 \cdot k^{2n} + (T_3 - T_1 - T_2) \cdot k^n + T_2.$$

(a) Berechnen Sie mittels der vorgegeben Vorschrift $1205 \cdot 6102$. Wenden Sie die Rekursionsvorschrift solange an, bis die zu multiplizierenden Zahlen einstellig sind.

(b) Überführen Sie die oben beschriebene Idee in einen Pseudocode.

(c) Bestimmen Sie eine rekursive Darstellung der Laufzeit dieses Algorithmus. Überführen Sie diese anschließend mittels des Master-Theorems in eine nicht-rekursive Form.

a) $1205 \cdot 6102 =$

$$= (\underbrace{12 \cdot 10^2}_{=1200} + 5) \cdot (61 \cdot 10^2 + 2)$$

$$= 12 \cdot 61 \cdot 10^4 + 12 \cdot 2 \cdot 10^2 + 5 \cdot 61 \cdot 10^2 + 5 \cdot 2$$

$$= \underbrace{12 \cdot 61}_{T_1} \cdot 10^4 + \underbrace{(12 \cdot 2 + 5 \cdot 61)}_{= \underbrace{(12+5) \cdot (61+2)}_{=: T_3} - \underbrace{12 \cdot 61}_{T_1} - \underbrace{5 \cdot 2}_{T_2}} \cdot 10^2 + \underbrace{5 \cdot 2}_{T_2}$$

$$T_1 = 12 \cdot 61 =$$
$$= (1 \cdot 10 + 2) \cdot (6 \cdot 10 + 1)$$
$$= 1 \cdot 6 \cdot 10^2 + (1 \cdot 1 + 2 \cdot 6) \cdot 10 + 1 \cdot 2$$
$$= \underbrace{1 \cdot 6}_{T_1'} \cdot 10^2 + [\underbrace{(1+2) \cdot (6+1)}_{T_3'} - \underbrace{1 \cdot 6}_{T_1'} - \underbrace{2 \cdot 1}_{T_2'}] \cdot 10 + \underbrace{1 \cdot 2}_{T_2'}$$
$$= 6 \cdot 10^2 + [3 \cdot 7 - 6 - 2] \cdot 10 + 2$$
$$= 6 \cdot 10^2 + [21 - 6 - 2] \cdot 10 + 2$$
$$= 600 + 13 \cdot 10 + 2$$
$$= \underline{\underline{732}}$$

$$T_3 = 17 \cdot 63 =$$
$$= T_1'' \cdot 10^2 + (T_3'' - T_1'' - T_2'') 10 + T_2''$$
$$= 1 \cdot 6 \cdot 10^2 + [(1+7)(6+3) - 1 \cdot 6 - 7 \cdot 3] \cdot 10 + 7 \cdot 3$$
$$= 600 + [72 - 6 - 21] \cdot 10 + 21$$
$$= 600 + 450 + 21$$
$$= 1071$$

$$= 732 \cdot 10^4 + (1071 - 732 - 10) \cdot 10^2 + 10$$
$$= 732 \cdot 10^4 + 329 \cdot 10^2 + 10$$
$$= 7320000 + 32900 + 10 = \underline{\underline{7352910}}$$

$$T_1 = a_h b_h, \quad T_2 = a_l b_l, \quad T_3 = (a_h + a_l)(b_h + b_l)$$

Laufzeit der Addition hängt linear von der Anzahl der Ziffern ab

net werden, verknüpft mittels Verschiebe- und Additionsope

irgendeine Konstante

$$a \cdot b = T_1 \cdot k^{2n} + (T_3 - T_1 - T_2) \cdot k^n + T_2.$$

Laufzeit

$$L(m) = \begin{cases} 3 \cdot L\left(\frac{m}{2}\right) + C \cdot m & \text{falls } m > 1 \\ 1 & \text{falls } m = 1 \end{cases}$$

Die Anzahl der Ziffern in a und b (d.h. $m = 2n$)

**Theorem**

*Seien $a \geq 1$, $b > 1$ und $d \in \mathbb{N}$ Konstanten, $f : \mathbb{N}_0 \to \mathbb{R}$ eine asymptotisch positive Funktion und die Funktion $T$ erfülle die Rekursionsgleichung*

$$T(n) = \begin{cases} \Theta(1) & \text{falls } n \leq d \\ aT(n/b) + f(n) & \text{sonst} \end{cases}$$

*wobei $n/b$ als entweder $\lfloor n/b \rfloor$ oder $\lceil n/b \rceil$ zu interpretieren ist.*
*Es sei $\ell := \log_b a$.*
*Dann kann $T$ asymptotisch wie folgt beschränkt werden:*

- *(1) Ist $f(n) = O(n^{\ell - \varepsilon})$ für ein $\varepsilon > 0$, so gilt $T(n) = \Theta(n^{\ell})$.*
- *(2) Ist $f(n) = \Theta(n^{\ell})$, so gilt $T(n) = \Theta(n^{\ell} \log n)$.*
- *(3) Ist $f(n) = \Omega(n^{\ell + \varepsilon})$ für ein $\varepsilon > 0$ und gilt $a \cdot f(n/b) \leq c \cdot f(n)$ für eine Konstante $c < 1$ und alle hinreichend großen $n$, so gilt $T(n) = \Theta(f(n))$.*

$a = 3, \quad b = 2$

$\ell = \log_2 3 \approx 1{,}58$

$f(m) = C \cdot m \in O(m) \subseteq O(m^{1{,}58 - \overset{\varepsilon}{0{,}1}})$

$\Rightarrow$ (1) trifft zu

$\Rightarrow \boxed{L(m) = \Theta(m^{\log_2(3)})}$

$\forall 0 < a \leq b$:
$O(n^a) \subseteq O(n^b)$

nicht rekursive Darstellung der Laufzeit

$\ell = 1$

$f(m) = O(m) \not\subseteq O(m^{1 - 0{,}001})$

Bsp.:

$O(n) \subseteq O(n^2)$

$O(n) \subseteq O(n^{1{,}5})$

$O(n^{100}) \subseteq O(n^{101})$

(d) Ein eifriger Student meint, dieses rekursive Verfahren verbessern zu können, indem er die Zahlen nun in jeweils vier möglichst gleich lange Teile aufteilt, statt in zwei. Seine neue Berechnung zeigt, dass er so (neben Additionen und Verschiebungen) sieben rekursive Multiplikationen braucht. Hat der Student mit seiner Behauptung recht?

d)

$$L(m) = \begin{cases} 7 \cdot L\left(\frac{m}{4}\right) + \underbrace{C \cdot m}_{f(m)} & \text{falls } m > 1 \\ 1 & \text{falls } m = 1 \end{cases}$$

$a = 7, \quad b = 4$

$\ell = \log_4 7 \approx 1{,}4$

$$f(m) = O(m) \leq O\left(m^{1{,}4 - \overbrace{0{,}00971}^{=\varepsilon > 0}}\right)$$

$$\underset{(1)}{\Rightarrow} \boxed{L(m) = \Theta\left(m^{\log_4 7}\right)}$$

Laufzeit ist besser geworden!

**Theorem**

*Seien $a \geq 1$, $b > 1$ und $d \in \mathbb{N}$ Konstanten, $f : \mathbb{N}_0 \to \mathbb{R}$ eine asymptotisch positive Funktion und die Funktion $T$ erfülle die Rekursionsgleichung*

$$T(n) = \begin{cases} \Theta(1) & \text{falls } n \leq d \\ aT(n/b) + f(n) & \text{sonst} \end{cases}$$

*wobei $n/b$ als entweder $\lfloor n/b \rfloor$ oder $\lceil n/b \rceil$ zu interpretieren ist.*
*Es sei $\ell := \log_b a$.*
*Dann kann $T$ asymptotisch wie folgt beschränkt werden:*

- *(1) Ist $f(n) = O(n^{\ell-\varepsilon})$ für ein $\varepsilon > 0$, so gilt $T(n) = \Theta(n^{\ell})$.*
- *(2) Ist $f(n) = \Theta(n^{\ell})$, so gilt $T(n) = \Theta(n^{\ell} \log n)$.*
- *(3) Ist $f(n) = \Omega(n^{\ell+\varepsilon})$ für ein $\varepsilon > 0$ und gilt $a \cdot f(n/b) \leq c \cdot f(n)$ für eine Konstante $c < 1$ und alle hinreichend großen $n$, so gilt $T(n) = \Theta(f(n))$.*

ealisiert werden kann, müssen hier im Wesentlichen nur noch jeweils drei Produkte

$$T_1 = a_h b_h, \qquad T_2 = a_l b_l, \qquad T_3 = (a_h + a_l)(b_h + b_l)$$

ekursiv berechnet werden, verknüpft mittels Verschiebe- und Additionsoperationen:

$$a \cdot b = T_1 \cdot k^{2n} + (T_3 - T_1 - T_2) \cdot k^n + T_2.$$

```
public int multiply(int a, int b) {
    int m = max{Anzahl Ziffer von a in Binär, Anzahl von Ziffer von b in Binär}
    if( m == 1 ){
        return a·b;
    }
    if(n ungerade){
        m=m+1;
    }
    int n = m/2
    int al = die niederwertigsten n Ziffer in Binär von a
    int al = die restlichen höcherwertige Ziffer von a
    int bh =
    int bl =
    int T1 = multiply(a , bl);
    int T2 = multiply(al, bl);
    int T3 = multiply(ah + al, bh+bl);
    return T1 << 2n + (T3 - T1 - T2) << n + T2;
}
```

$T1 \cdot k^{2n}$ h

$10100 \ll 3$

$\downarrow$

$10100000$