Prof. Dr. Ir. Joost-Pieter Katoen

Christian Dehnert, Friedrich Gretz, Benjamin Kaminski, Thomas Ströder

Tutoraufgabe 1 (Rekursionsgleichungen):

Geben Sie die Rekursionsgleichung $T(d, l)$ für die Laufzeit des folgenden Algorithmus an. Gehen Sie dabei davon aus, dass die elementaren Operationen $\{+, -, \cdot, /, \sqrt{}\} \in \Theta(1)$ liegen.

```
float sierpinski(int depth, float length) {
  if (depth > 0) {
    return  sierpinski(depth-1, length/2) +
            sierpinski(depth-1, length/2) +
            sierpinski(depth-1, length/2);
  } else {
    // berechne die Flaeche des Dreiecks
    return sqrt(length^2 - (length/2)^2) * (length/2);
  }
}
```

Hinweis: `http://de.wikipedia.org/wiki/Sierpinski-Dreieck`

Lösung:

Die Rekursionsgleichung lautet

$$T(d, l) = \begin{cases} 3 \cdot T(d-1, \frac{l}{2}) + (6+2) & \text{falls } d > 0 \\ 7 & \text{sonst} \end{cases}$$

Der Parameter l hat also keinen Einfluss auf die Laufzeit.

Tutoraufgabe 2 (Substitutionsmethode):

Gegeben sei die folgende Rekursionsgleichung:

$$T(n) \quad = \quad \begin{cases} 4T(\frac{n}{2}) + \frac{n^2}{\log_2(n)}, & \text{falls } n > 1 \\ 1, & \text{sonst} \end{cases}$$

- Schätzen Sie mit Hilfe des Rekursionsbaumes eine (möglichst asymptotisch scharfe) obere Schranke für die Komplexitätsklasse der Laufzeit $T(n)$, d.h., geben Sie eine nicht-rekursive Funktion $f(n)$ mit $T(n) \in \mathcal{O}(f(n))$ an.
- Beweisen Sie mit Hilfe der Substitutionsmethode, dass $T(n) \in \mathcal{O}(f(n))$.

Hinweis: Die n–te Partialsumme der harmonischen Reihe ist definiert durch $H_n := \sum_{i=1}^{n} \frac{1}{i}$ und hat folgende Eigenschaften:

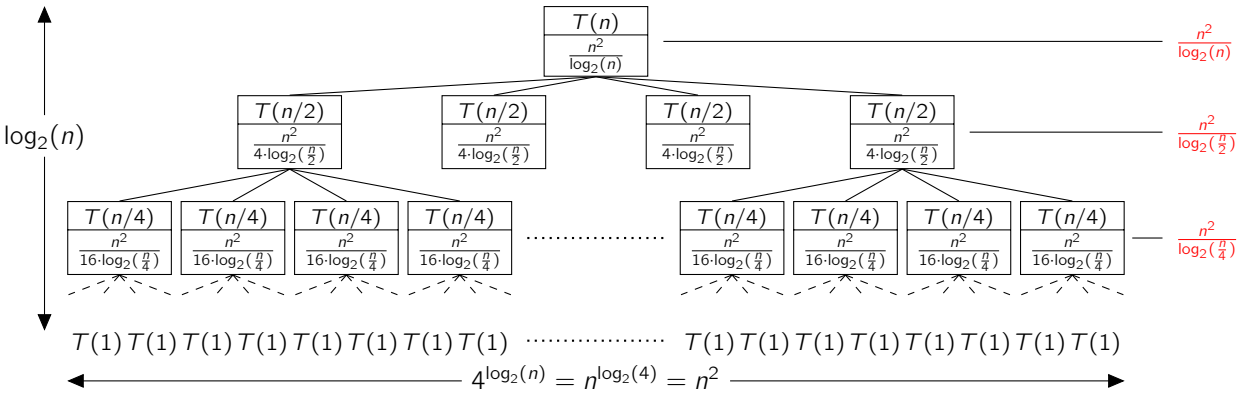
(i) $\forall n \geq 1.\ \log(n) < H_n$

(ii) $\forall n > 1.\ H_n - \log(n) \ < \ H_{n-1} - \log(n-1)$

Lösung:

Hinweis:
Das Mastertheorem ist nicht anwendbar, da $\frac{n^2}{\log_2(n)} \notin \mathcal{O}(n^{2-\epsilon})$, $\frac{n^2}{\log_2(n)} \notin \Theta(n^2)$ *und* $\frac{n^2}{\log_2(n)} \notin \Omega(n^{2+\epsilon})$.

Wir betrachten den Rekursionsbaum:



Aus dem Rekursionsbaum leiten wir folgende Abschätzung ab:

$$\begin{aligned}
T(n) &= \left(\sum_{i=0}^{\log_2(n)-1} \frac{n^2}{\log_2(\frac{n}{2^i})}\right) + n^2 \\
&= n^2 \cdot \left(\sum_{i=0}^{\log_2(n)-1} \frac{1}{\log_2(n)-\log_2(2^i)}\right) + n^2 \\
&= n^2 \cdot \left(\sum_{i=0}^{\log_2(n)-1} \frac{1}{\log_2(n)-i}\right) + n^2 \\
&= n^2 \cdot \left(\sum_{i=1}^{\log_2(n)} \frac{1}{i}\right) + n^2 && \text{(Summe gedreht)} \\
&\approx n^2 \cdot \log(\log_2(n)) + n^2 && \text{(mit wachsendem } n\text{)}
\end{aligned}$$

Daher stellen wir folgende Vermutung auf:

$$T(n) \in \mathcal{O}(n^2 \cdot \log(\log_2(n)))$$

Behauptung: $T(n) \in \mathcal{O}(n^2 \cdot \log(\log_2(n)))$

Beweis:
Wir müssen zeigen, dass:

$$\exists c_1 > 0.\ \exists n_0 \geq 0.\ \forall n \geq n_0.\ T(n) \leq c_1 \cdot n^2 \cdot \log(\log_2(n)))$$

Wähle $n_0 = 4$ und $c_1 = 10 \cdot \frac{1}{\log 2}$.

Induktionsanfang:

- $T(4) = 4 \cdot T(2) + \frac{4^2}{\log_2 4} = 4 \cdot \underbrace{(4 \cdot T(1) + \frac{2^2}{\log_2 2})}_{T(2)=8} + \frac{4^2}{\log_2 4} = 40$
  $\leq 10 \cdot \frac{1}{\log 2} \cdot 4^2 \cdot \log(\log_2(4)) = 10 \cdot 4^2 = 160$
- $T(5) = 4 \cdot T(3) + \frac{5^2}{\log_2 5} = 4 \cdot (4 \cdot T(2) + \frac{3^2}{\log_2 3}) + \frac{5^2}{\log_2 5} \approx 4 \cdot (4 \cdot 8 + 5.68) + 10.77 = 161.49$
  $\leq 10 \cdot \frac{1}{\log 2} \cdot 5^2 \cdot \log(\log_2(5)) \approx 303.83$
- $T(6) = 4 \cdot T(3) + \frac{6^2}{\log_2 6} = 4 \cdot (4 \cdot T(2) + \frac{3^2}{\log_2 3}) + \frac{6^2}{\log_2 6} \approx 4 \cdot (4 \cdot 8 + 5.68) + 13.93 = 164.65$
  $\leq 10 \cdot \frac{1}{\log 2} \cdot 6^2 \cdot \log(\log_2(6)) \approx 493.25$
- $T(7) = 4 \cdot T(4) + \frac{7^2}{\log_2 7} \approx 4 \cdot 40 + 17.45 = 177.45$
  $\leq 10 \cdot \frac{1}{\log 2} \cdot 7^2 \cdot \log(\log_2(7)) \approx 729.71$

Induktionsvoraussetzung: $\forall n_0 \leq m < n.\ T(m) \leq c_1 \cdot m^2 \cdot \log(\log_2(m))$
Induktionsschluss:

$$
\begin{aligned}
T(n) &= 4 \cdot T(\tfrac{n}{2}) + \tfrac{n^2}{\log_2(n)} \\
&\leq 4 \cdot c_1 \cdot \tfrac{n^2}{4} \cdot \log(\log_2(\tfrac{n}{2})) + \tfrac{n^2}{\log_2(n)} && \text{Induktionsvoraussetzung} \\
&= c_1 \cdot n^2 \cdot \log(\log_2(n) - \log_2(2)) + \tfrac{n^2}{\log_2(n)} \\
&= c_1 \cdot n^2 \cdot \log(\log_2(n) - 1) + \tfrac{n^2}{\log_2(n)} \\
&< c_1 \cdot n^2 \cdot (\log(\log_2(n)) - \tfrac{1}{\log_2(n)}) + \tfrac{n^2}{\log_2(n)} && \text{(ii)} \Leftrightarrow \log(k-1) < H_{k-1} - H_k + \log(k) = -\tfrac{1}{k} + \log(k) \\
&= c_1 \cdot n^2 \cdot \log(\log_2(n)) - \tfrac{c_1 \cdot n^2}{\log_2(n)} + \tfrac{n^2}{\log_2(n)} \\
&= c_1 \cdot n^2 \cdot \log(\log_2(n)) + \underbrace{(1 - c_1) \cdot (\frac{n^2}{\log_2(n)})}_{<0 \text{ für } c_1=4 \text{ und } n \geq n_0=4} \\
&\leq c_1 \cdot n^2 \cdot \log(\log_2(n))
\end{aligned}
$$

□

Tutoraufgabe 3 (Variablentransformation):

Nutzen Sie die Methode der Variablentransformation um eine (asymptotisch scharfe) obere Schranke für die Rekursionsgleichung

$$T(n) = T(\sqrt[3]{n}) + 3$$

zu bestimmen.

Lösung:

a)

$$
\begin{aligned}
& T(n) = T(\sqrt[3]{n}) + 3 && \text{| Variablentransformation } m = \log_2 n \\
\Rightarrow\ & T(2^m) = T(2^{\frac{m}{3}}) + 3 && \text{| Umbenennung } T(2^m) = S(m) \\
\Rightarrow\ & S(m) = S(\frac{m}{3}) + 3 && \text{| Lösung Rekursionsgleichung (z.B. MT): } S(m) = S(\frac{m}{3}) + 3 \\
\Rightarrow\ & S(m) \leq c \cdot \log_3 m && \text{| } m = \log_2 n \\
\Rightarrow\ & T(n) \leq c \cdot \log_3 \log_2 n \\
\Rightarrow\ & T(n) \in \mathcal{O}(\log \log_2 n))
\end{aligned}
$$

Tutoraufgabe 4 (Master-Theorem):

Geben Sie für folgende Rekursionsgleichungen (mit Begründung) an, ob diese mit dem Master-Theorem gelöst werden können. Falls dies der Fall ist, geben Sie auch die resultierende Komplexitätsklasse an.

a) $T(n) = 4 \cdot T\left(\frac{n}{2}\right) + 5n$

b) $T(n) = 4 \cdot T\left(\frac{n}{4}\right) + 4n$

c) $T(n) = 2 \cdot T\left(\frac{n}{2}\right) + n^2$

d) $T(n) = 2 \cdot T(n)$

Lösung:

a) Es sind $b = 4$, $c = 2$ und $f(n) = 5n$. E wird nun wie folgt bestimmt:

$$E \;=\; \frac{\log(4)}{\log(2)} \;=\; \frac{2}{1} \;=\; 2$$

Damit gilt $n^E = n^2$. Wähle nun $\varepsilon = 1$. Damit gibt es nun zwei Konstanten $c' = 5$ und $n_0 = 1$, sodass gilt

$$\forall n \geq n_0 \colon f(n) \;=\; 5n \;\leq\; 5 \cdot n^{2-1} \;=\; c' \cdot n^{E-\varepsilon}$$

und damit gilt $f(n) \in \mathcal{O}(n^{E-\varepsilon})$. Somit findet der *erste Fall* des Master-Theorems Anwendung und es ergibt sich die Komplexitätsklasse $\Theta\left(n^E\right)$ für $T(n)$, also

$$T(n) \in \Theta(n^2)\ .$$

b) Es sind $b = 4$, $c = 4$ und $f(n) = 4n$. E wird nun wie folgt bestimmt:

$$E \;=\; \frac{\log(4)}{\log(4)} \;=\; \frac{2}{2} \;=\; 1$$

Damit gilt $n^E = n$. Damit gibt es nun drei Konstanten $c_1 = c_2 = 4$ und $n_0 = 1$, sodass gilt

$$\forall n \geq n_0 \colon c_1 \cdot n^E \;=\; 4 \cdot n^1 \;\leq\; \underbrace{4n}_{=f(n)} \;\leq\; 4 \cdot n^1 \;=\; c_2 \cdot n^E$$

und damit gilt $f(n) \in \Theta\left(n^E\right)$. Somit findet der *zweite Fall* des Master-Theorems Anwendung und es ergibt sich die Komplexitätsklasse $\Theta\left(n^E \cdot \log(n)\right)$ für $T(n)$, also

$$T(n) \in \Theta\big(n \cdot \log(n)\big)\ .$$

c) Es sind $b = 2$, $c = 2$ und $f(n) = n^2$. E wird nun wie folgt bestimmt:

$$E \;=\; \frac{\log(2)}{\log(2)} \;=\; \frac{1}{1} \;=\; 1$$

Damit gilt $n^E = n$. Wähle nun $\varepsilon = 1$. Damit gibt es nun zwei Konstanten $c' = 1$ und $n_0 = 1$, sodass gilt

$$\forall n \geq n_0 \colon f(n) \;=\; n^2 \;\geq\; 1 \cdot n^{1+1} \;=\; c' \cdot n^{E+\varepsilon}$$

und damit gilt $f(n) \in \Omega\left(n^{E+\varepsilon}\right)$. Somit fände der *dritte Fall* des Master-Theorems Anwendung. Es ist jedoch noch die zweite Bedingung $\exists\, 0 \leq d < 1\ \exists\, n_0\ \forall n \geq n_0 \colon b \cdot f\left(\frac{n}{c}\right) \leq d \cdot f(n)$ zu überprüfen. Wir wählen dafür $d = \frac{1}{2}$. Damit ergibt sich

$$b \cdot f\left(\frac{n}{c}\right) \;=\; 2 \cdot \frac{n^2}{2^2} \;=\; \frac{1}{2} \cdot n^2 \;\leq\; \frac{1}{2} \cdot n^2 \;=\; d \cdot f(n)\ ,$$

was offensichtlich für alle n gilt. Somit sind beide Bedingungen für den dritten Fall des Master-Theorems erfüllt und es ergibt sich die Komplexitätsklasse $\Theta\big(f(n)\big)$ für $T(n)$, also

$$T(n) \in \Theta(n^2)\ .$$

d) Es sind $b = 2$, $c = 1$ und $f(n) = 0$. E wird nun wie folgt bestimmt:

$$E = \frac{\log(2)}{\log(1)} = \frac{2}{0} = \textit{undefiniert}$$

Damit kann das Master-Theorem *nicht* angewendet werden.

Ein Algorithmus, dessen Laufzeit durch die vorliegende Rekursionsgleichung beschrieben werden kann, wäre beispielsweise der folgende:

```
int diverge(int k) {
    if (k > 0){
        diverge(k);
        diverge(k);
    }
}
```

Die Rekursionsgleichung $T(0) = 0$, $T(n) = 2 \cdot T(n)$ beschreibt die Anzahl der Aufrufe von `diverge`. Dieser Algorithmus terminiert jedoch für alle $k > 0$ nicht. Der Grund dafür ist, dass jeder Aufruf von `diverge` die Funktion `diverge` wieder mit demselben k aufgeruft. k wird also mit jedem Aufruf *nicht* verringert. Die „Komplexitätsklasse“ des Algorithmus' ist sozusagen $\Theta(\infty)$.