Gruppennummer: 22

Datenstrukturen und Algorithmen, Abgabe 4

Beccard, Vincent, 377979; Braun, Basile 388542; Brungs, Simon, 377281

9. Mai 2018

1 Trinäre Suche

Wir betrachten die Größe des Arrays im schlimmsten Fall, bei einer erfolglosen Suche. Dieser tritt ein, wenn die letzte else - Bedingung (im Code ab Zeile 14) ausgeführt wird. Sei nun im Folgenen l = left und r = right.
$lmid - l = \lceil (2l+3)/3 \rceil - l = \lceil (r-l)/3 \rceil = \lceil (n-1)/3 \rceil$
oder:
$r - rmid = r - \lfloor (l-2r)/3 \rfloor = \lceil (r-l)/3 \rceil = \lceil (n-1)/3 \rceil$
Im schlimmsten Fall ist die neue Größe des Arrays also: $\lceil (n-1)/3 \rceil$
Daraus ergibt sich folgende Rekursionsgleichung:

$$S(n) = \begin{cases} 0 & \text{für } n = 0 \\ 1 + S(\lceil (n-1)/2 \rceil) & \text{für } n > 0 \end{cases}$$

Da der Spezialfall n = $\frac{3^k-1}{2}$ lässt sich daraus herleiten: $\lceil (n-1)/3 \rceil = \lceil (\frac{3^k-1}{2} - 1)/3 \rceil = \lceil (3^k-3)/6 \rceil = \lceil 3^{k-2} - \frac{1}{2} \rceil = 3^{k-2}$
Daher gilt für k ¿ 0 nach der Definition $S(n) = 1 + S(\lceil (n-1)/2 \rceil)$, dass S($\frac{3^k-1}{2}$) $= 1 + S(3^{k-2})$ und damit S($\frac{3^k-1}{2}$) = k - 2
Vermutung: $S(3^k) = 1 + S(3^{k-2})$.
$S(n)$ steigt monoton, da für n=0 S(n)=0, n=1 S(n)=1, n=2 S(n)=2, n=3 S(n)=2, n=4 S(n)=2, n=5 S(n)=3 etc. Also ist $S(n) = k - 2$, falls $3^k <= n < 3^{k-2}$.
Oder: falls $k <= log_3(n) < k - 2$.
Dann ist $S(n) = \lfloor log_3(n) \rfloor + 1$

2 Bilineare Suche

a Worst-Case Laufzeit$W(n)$

Die Worst-Case Laufzeit ist $W(n) = n-1$, da beide Seiten das Element K finden müssen. Dies dauert umso länger, desto weiter das Element vom Ende/Anfang entfernt ist. Somit ist der schlimmste Fall wenn das Element am Ende/Anfang liegt, da so damit der weiter entfernte Zeiger erst einmal über alle anderen Elemente drüber laufen muss, also über $n - 1$ Elemente.

Beccard, Vincent, 377979; Braun, Basile 388542; Brungs, Simon, 377281;
Nummer der Übungsgruppe: 22

b Best-Case Laufzeit $B(n)$

Die Best-Case Laufzeit ist $B(n) = \lfloor \frac{n}{2} \rfloor$, da damit der Weg von beiden Seiten des Arrays am kürzesten ist das Element in der Mitte liegen muss. Somit müssen die Zeiger jeweils die Hälfte des Arrays durchlaufen, um das Element zu finden.

c Average-Case Laufzeit $A(n)$

Die Average-Case Laufzeit ist $A(n) = \frac{\sum_{i=1}^{\lceil \frac{n}{2} \rceil - 1} n - i}{\lceil \frac{n}{2} \rceil} + \frac{\lfloor \frac{n}{2} \rfloor}{\lceil \frac{n}{2 + 2 * (n \pmod 2)} \rceil}$.
Im ersten Teil der Formel, wird die Dauer für die Hälfte aller möglichen Fälle, ohne den Best-Case, summiert, und dann durch die Hälfte der Anzahl der Fälle geteilt, so dass man die Durchschnittslaufzeit ohne den Best-Case für die Hälfte der Fälle erhält, die gleich der Durchschnittslaufzeit ohne den Best-Case für alle Fälle ist. Im nächsten Teil der Formel wird dann noch die Laufzeit für den Best-Case hinzu addiert, geteilt durch die Wahrscheinlichkeit, dass dieser Auftritt. Somit erhält man die Formel für die Average-Case Laufzeit.

3 Substitionsmethode Reloaded

a

zZ: Jede Teilmenge M von $\mathbb{T}$ hat eine kleinste obere und eine größte untere Schranke.
S,T $\in$ M $\subseteq \mathbb{T}$
S $\preceq$ T $\leftrightarrow \forall$n: $S(n) <= T(n)$—¡= ist eine Totalordnung auf $\mathbb{R}$
$\Rightarrow \forall$R,T: R $\preceq$ T $\vee$ T $\preceq$ R
$\Rightarrow \exists$S $\succeq$ T $\in$ M : (T $\preceq$ S) $\wedge$ ($\forall$Y $\in$ M : T $\preceq$ y $\Rightarrow$ S $\preceq$ Y)
$\Rightarrow$ Jede Teilmene M von $\mathbb{T}$ hat eine kleinste obere Schranke.

S,T $\in$ M $\subseteq \mathbb{T}$
S $\preceq$ T $\leftrightarrow \forall$n: $S(n) <= T(n)$—¡= ist eine Totalordnung auf $\mathbb{R}$
$\Rightarrow \forall$R,T: R $\preceq$ T $\vee$ T $\preceq$ R
$\Rightarrow \exists$S $\preceq$ T $\in$ M : (T $\succeq$ S) $\wedge$ ($\forall$Y $\in$ M : T $\succeq$ y $\Rightarrow$ S $\succeq$ Y)
$\Rightarrow$ Jede Teilmene M von $\mathbb{T}$ hat eine größte untere Schranke.

Somit hat jede Teilmenge M von $\mathbb{T}$ eine kleinste untere- und eine größte obere Schranke im Sinne der Halbordnung $\preceq$. Somit ist $(\mathbb{T}, \preceq)$ ein vollständiger Verband.

b

zZ: X $\preceq$ Y $\Rightarrow \Psi$(X) $\preceq \Psi$(Y)
$\Leftrightarrow$ X $\preceq$ Y $\Rightarrow$ 2X($\lfloor$n/2$\rfloor$) + n ¡= 2Y($\lfloor$n/2$\rfloor$) + n

Fall 1: n = 0 X $\preceq$ Y $\Rightarrow$ 0 ¡= 0 ist eine wahre Aussagen.
Fall 2: n $\in \mathbb{N}$ X $\preceq$ Y $\Rightarrow$ 2X($\lfloor$n/2$\rfloor$)+ n ¡= 2Y($\lfloor$n/2$\rfloor$)+ n — n $\in \mathbb{N}$ sei c:= ($\lfloor$n/2$\rfloor$
$\Leftrightarrow$ X $\preceq$ Y $\Rightarrow$ 2X(c) + n ¡= 2Y(c) + n — -n
$\Leftrightarrow$ X $\preceq$ Y $\Rightarrow$ 2X(c)¡= 2Y(c) — :2
$\Leftrightarrow$ X $\preceq$ Y $\Rightarrow$ X(c)¡= Y(c) ist eine warhe Aussage. Somit ist gezeigt, dass Ψ monoton bezüglich $\preceq$ ist.

c

d

e

f

g

Die Aufgabe e) fiel mir leichter, da die Methode in der Vorlesung vorgestellt wurde.

4 Rekursionsbäume

a

Gegeben: T(0) = 1
T(1) = 1
T(n) = 4 * T($\lfloor n/16 \rfloor$) + n