Fakultät für Informatik
Lehrstuhl 11
Prof. Dr. Kevin Buchin
Dr. Carolin Rehs
Antonia Kalb
Erik Thordsen

Übungen zur Vorlesung

Datenstrukturen, Algorithmen und Programmierung 2

Sommersemester 2024

Übungsblatt 3

Besprechungszeit:
6.5. – 10.5.2024

Aufgabe 3.1 – Landau-Symbole (5 Punkte)

Füllen Sie die folgende Tabelle aus. Geben Sie jeweils an, ob gilt:

$\mathcal{O}$: $f \in \mathcal{O}(g)$ aber $f \notin \Omega(g)$

Ω: $f \in \Omega(g)$ aber $f \notin \mathcal{O}(g)$

Θ: $f \in \Theta(g)$.

Orientieren Sie sich dabei an der ersten Zeile. Es müssen keine Begründungen angegeben werden.

f \ g	$\frac{1}{n}$	$\log n$	n	$n \log n$	n^2	$n!$
$2n \log n - 13 \log n$	Ω	Ω	Ω	Θ	$\mathcal{O}$	$\mathcal{O}$
$7n^2 + \sqrt[4]{n^6}$						
$2^{\log_2(n)}$						
$10n + (\sqrt[4]{n})^6$						
2^{n+2}						
$\sum_{i=1}^{n} \frac{1}{i}$						

Aufgabe 3.2 – Rekursionsgleichung (15 Punkte)

Gegeben sei die Rekursionsgleichung:

$$T(n) = \begin{cases} 4 \cdot T(n/2) + n^3 & \text{wenn } n > 1 \\ 1 & \text{sonst} \end{cases}$$

a) Bestimmen Sie eine Funktion $g(n)$ und beweisen Sie $T(n) \in O(g(n))$ mittels Rekursionsbaummethode.

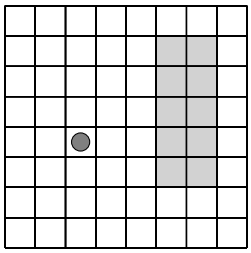
b) Bestimmen Sie eine Funktion $g(n)$ und beweisen Sie $T(n) \in O(g(n))$ mittels Substitutionsmethode.
Tipp: Sie können hier für $g(n)$, die gleiche Funktion nehmen, die Sie in Aufgabenteil a) bestimmt haben.

c) Bestimmen Sie eine Funktion $g(n)$ und beweisen Sie $T(n) \in \Theta(g(n))$ mittels Mastermethode.

Aufgabe 3.3 – (Teile und Herrsche) (20 Punkte)

Auf unserem Gelände wurde eingebrochen! Zum Glück haben wir ein Sicherheitssystem mit Bewegungsmeldern. Der Einfachheit halber haben wir unser quadratisches Gelände in kleine Quadrate aufgeteilt, sodass jedes Quadrat einen Bewegungsmelder hat.

Wir nennen die Seitenlänge n und nehmen an, dass n eine Zweierpotenz ist. Im folgenden Beispiel ist $n = 8$:



Die Sicherheitssoftware hat gemeldet, dass ein Bewegungsmelder ausgelöst wurde, aber nicht welcher. Die Software hat bislang leider keine Suchfunktion, sodass wir nicht direkt abfragen können, wo ein Bewegungsmelder ausgelöst wurde, also wo sich die einbrechende Person gerade befindet. Sie hat jedoch eine Funktion, um zu überprüfen, ob in einem bestimmten Rechteck ein Bewegungsmelder ausgelöst wurde. Oben ist ein Rechteck eingezeichnet, das wir anfragen könnten, und in dem der Bewegungsmelder nicht ausgelöst wurde.

Ein Rechteck kann man zum Beispiel durch die linke untere und rechte obere Ecke beschreiben. Eine Anfage an die Sicherheitssoftware kann man also durch zwei Ecken (i_1, j_1) und (i_2, j_2) mit $1 \leq i_1 \leq i_2 \leq n$ und $1 \leq j_1 \leq j_2 \leq n$ ausdrücken. Die Antwort ist dann *ja*, wenn sich die Person an einer Position (i, j) mit $i \in \{i_1, \ldots, i_2\}$ und $j \in \{j_1, \ldots, j_2\}$ befindet, und *nein* sonst. Wir möchten nun die Person mit möglichst wenigen Anfragen an die Softare finden.

a) Entwerfen Sie einen Teile-und-Herrsche-Algorithmus mit Namen `PersonenSuche`, der die einbrechende Person findet, und beschreiben Sie ihn mit eigenen Worten. Der Algorithmus soll mit $\mathcal{O}(\log n)$ Anfragen an die Software auskommen und die Person immer finden.

b) Geben Sie eine Implementierung Ihres Algorithmus in Pseudocode an. Die entsprechende Funktion der Sicherheitssoftware kann dabei mit `Bewegungsmeldung`(i_1, j_1, i_2, j_2) aufgerufen werden und antwortet mit 1, wenn sich die Person im angegebenen Rechteck versteckt, und mit 0, wenn die Person sich nicht dort versteckt.

c) Beweisen Sie die Korrektheit Ihres Algorithmus.

d) Zeigen Sie, dass Ihr Algorithmus die Laufzeit $\mathcal{O}(\log n)$ hat, wenn wir davon ausgehen, dass jeder Aufruf der Funktion `Bewegungsmeldung`(i_1, j_1, i_2, j_2) konstante Laufzeit 1 hat.