Tutorium
Grundlagen: Algorithmen und Datenstrukturen

Übungsblatt Woche 2

Organisatorisches

Leistungsnachweis

- Klausur (voraussichtlich) 9. Aug. 2023
  - schriftliche Prüfung
  - Dauer: 90 Minuten
  - Erlaubtes Hilfsmittel: hand-beschriebenes DIN A4 Blatt
- Wiederholungs-Klausur (voraussichtlich) Oktober 2023
- Vorbereitung durch aktive Teilnahme an Vorlesung und Übungsbetrieb
- Notenbonus von 0,3 auf bestandene Klausur möglich durch Hausaufgaben

6

Aufgabe 2.1 – Division

Aufgabe 2.1 (P) Division

Wir betrachten die Schulmethode der Division. Hierbei untersuchen wir die Zahl der (elementaren) Operationen, die wir benötigen, um eine n-stellige Zahl durch eine einzelne *Ziffer* ganzzahlig mit Rest zu teilen. Für zwei Zahlen $a \in \mathbb{N}_0$ und $b \in \mathbb{N}$ bezeichne $g(a,b) := \lfloor a/b \rfloor$ das Ergebnis der ganzzahligen Division von a durch b, sowie $r(a,b) := a - g(a,b) \cdot b$ den Rest der ganzzahligen Division. Weiterhin bezeichne $a \oplus b$ die Konkatenation zweier Ziffern a und b.

Für die Auswertung von $g(x_i \oplus x_{i+1}, y)$ und $r(x_i \oplus x_{i+1}, y)$ verwenden wir im Voraus berechnete Tabellen. Wir *legen fest*, dass in unserem Modell jeder Vergleich zweier Zahlen, jede Konkatenation zweier Ziffern, jede Zuweisungsoperation, jedes In- und Dekrement, sowie jede Nachschlageoperation in den Tabellen eine *Grundoperation* darstellt.

Wie viele Grundoperationen hat die Schulmethode in unserem Rechenmodell im schlimmsten Fall, wenn man eine Zahl mit n Ziffern ganzzahlig durch eine Ziffer teilt? Betrachten Sie bei Ihrer Lösung jeweils jede Zeile im Pseudocode.

Aufgabe 2.1

Input: Ziffer[] $(x_1, x_2, \ldots, x_n)$, Ziffer y

int $i := 1$
Ziffer $x_0 := 0$ /* Hilfsziffer zur Vermeidung von Fallunterscheidung */
while $i \leq n$ **do**
    if $x_{i-1} > 0$ **then**
        Ziffer $z_i := g(x_{i-1} \oplus x_i, y)$
        $x_i := r(x_{i-1} \oplus x_i, y)$
        $x_{i-1} := 0$ /* Kann auch weggelassen werden */
    end
    else
        Ziffer $z_i := g(x_i, y)$
        $x_i := r(x_i, y)$
    end
    $i := i + 1$
end
Das Ergebnis der Division ist $z_1 \oplus z_2 \ldots$, der Rest is x_n.

Algorithm 1: Ganzzahlige Division mit Rest

Aufgabe 2.2 – Induktion

Beweisen Sie folgende Aussagen durch Induktion für alle $n \in \mathbb{N} := \{1, 2, 3, 4, 5, \dots\}$:

(a) $n^{\frac{n}{2}} \leq n!$

Aufgabe 2.2

(b) $19|(5 \cdot 2^{3n+1} + 3^{3n+2})$ (In Worten: 19 teilt $(5 \cdot 2^{3n+1} + 3^{3n+2})$ ganzzahlig)

Aufgabe 2.3 – Landau-Notation

Kreuzen Sie in den Zeilen (a) bis (d) jeweils das zu den Funktionen am besten passende und stärkste Symbol an. Δ ist dabei ein Platzhalter für die Möglichkeiten $o, \mathcal{O}, \omega, \Omega, \Theta$, und u. Das heißt, wenn $\Delta = o$ (bzw. $\Delta = \Theta$) möglich ist, wählen Sie $\Delta = o$ (bzw. $\Delta = \Theta$) und nicht $\Delta = \mathcal{O}$. Falls die Funktionen mit keinem dieser 5 Symbole vergleichbar sind, kreuzen Sie „u." an.

Bsp.:	n	$\in \Delta(n^2)$	☒ o	☐ $\mathcal{O}$	☐ ω	☐ Ω	☐ Θ	☐ u.
(a)	$7n^3$	$\in \Delta(3n^7)$	☐ o	☐ $\mathcal{O}$	☐ ω	☐ Ω	☐ Θ	☐ u.
(b)	$\sqrt[3]{\sqrt{3n}}$	$\in \Delta\left(\sqrt{\sqrt[3]{2n}}\right)$	☐ o	☐ $\mathcal{O}$	☐ ω	☐ Ω	☐ Θ	☐ u.
(c)	$n!$	$\in \Delta(4^n)$	☐ o	☐ $\mathcal{O}$	☐ ω	☐ Ω	☐ Θ	☐ u.
(d)	n	$\in \Delta((2 + (-1)^n)n)$	☐ o	☐ $\mathcal{O}$	☐ ω	☐ Ω	☐ Θ	☐ u.

Aufgabe 2.3

(a) $7n^3 \quad \in \Delta(3n^7)$ $\qquad \square\, o \quad \square\, \mathcal{O} \quad \square\, \omega \quad \square\, \Omega \quad \square\, \Theta \quad \square$ u.

(b) $\sqrt[3]{\sqrt{3n}} \in \Delta\left(\sqrt{\sqrt[3]{2n}}\right)$ $\qquad \square\, o \quad \square\, \mathcal{O} \quad \square\, \omega \quad \square\, \Omega \quad \square\, \Theta \quad \square$ u.

Aufgabe 2.3

(c) $n! \quad \in \Delta(4^n)$ $\quad \square\, o \quad \square\, \mathcal{O} \quad \square\, \omega \quad \square\, \Omega \quad \square\, \Theta \quad \square$ u.

(d) $n \quad \in \Delta((2+(-1)^n)n)$ $\quad \square\, o \quad \square\, \mathcal{O} \quad \square\, \omega \quad \square\, \Omega \quad \square\, \Theta \quad \square$ u.

Aufgabe 2.4

Sehen Sie sich die folgenden Funktionen an:

Funktion 1

```
int f(int[] A, int[] B) {
  int result = 0;
  for (int a : A) {
    for (int b : B) {
      if (a == b) {
        result += a;
        break;
      }
    }
  }
  return result;
}
```

Funktion 2

```
int f(int n) {
  if (n <= 0) return 0;
  if (n == 1) return 1;
  return f(n-1) + f(n-2);
}
```

Funktion 3

```
int f(int n) {
  if (n < 0) return -1;
  if (n == 0) return 1;
  return n * f(n-1);
}
```

(a) Erklären Sie in natürlicher Sprache, was die Funktionen jeweils berechnen.

(b) Welche asymptotischen Laufzeiten haben die Algorithmen? Begründen Sie Ihre Antwort.