Aufgabensammlung ADS-Repetitorium WS 25/26

Rekursive Algorithmen – Randomisierung – Hashing

Aufgabe 1: Rekursive Algorithmen

Geben Sie jeweils einen Algorithmus an, der die gegebene Laufzeit erfüllt. Alle Algorithmen sollen im Basisfall, also für $n = 1$, konstante Laufzeit haben. Als Eingabe bekommen die Algorithmen jeweils ein Feld der Länge n. Die Algorithmen sollten eine Rückgabe haben, doch der Wert der Rückgabe ist egal. Sie dürfen Rundungsfehler ignorieren.

(a) $T(n) = 3T(n-1) + n$

(b) $T(n) = 4T(n/4) + O(n)$

(c) $T(n) = T(\lceil n/2 \rceil) + T(\lfloor n/2 \rfloor) + T(\sqrt{n})$; bitte beachten Sie hier die Rundungen.

(d) $T(n) = 3T(n-5) + n \log n$

(e) $T(n) = T(n-1) + T(n-2) + 42T(n/2) + n^2$

Aufgabe 2: Rekursive Laufzeiten

Finden Sie für die nachstehenden Rekursionsgleichungen jeweils eine Funktion f, für die $T \in \Theta(f)$ gilt. Sie können davon ausgehen, dass die Laufzeit im Basisfall konstant ist.

(a) $T(n) = 4T(\lfloor n/2 \rfloor) + \frac{1}{2}n^2\sqrt{n}$

(b) $T(n) = 4T(\lfloor n/2 \rfloor) + n^2 \log n + n$

(c) $T(n) = T(n-3) + 2n$

(d) $T(n) = 2T(\lfloor n/4 \rfloor) + 3\sqrt{n}$

(e) $T(n) = 3T(\lfloor n/2 \rfloor) + \frac{n}{6}$

(f) $T(n) = 3T(\lfloor n/5 \rfloor) + \frac{1}{2}\sqrt{n}$

(g) $T(n) = 12T(\lfloor n/2 \rfloor) + n^4$

(h) $T(n) = T(\lfloor n/2 \rfloor) + T(\lceil n/2 \rceil) + \Theta(n \log n)$

Aufgabe 3: Von Monte Carlo nach Las Vegas

(a) Geben Sie eine scharfe obere Schranke für die Fehlerwahrscheinlichkeit von Algorithmus 6 an.

Algorithmus 6: `FindLarge`(**int**[] A, **int** k)

input : Array A ist ein Feld ganzzahliger paarweise verschiedener Zahlen
output: Gibt ein Element zurück, dass mindestens so groß ist wie der Median

```
1 m = 0
2 for i = 1 to k do
3 |   randomly choose r ∈ {1, ..., n}
4 |   if A[r] > m then
5 |   |   m = A[r]
6 return m
```

(b) Geben Sie die Erfolgswahrscheinlichkeit mit jedem Schritt von Algorithmus 7 an.

Algorithmus 7: `FindRepeated`(**int**[] A)

input : Array A der Länge n mit ganzzahligen Zahlen, sodass $\lceil n/2 \rceil$ Zahlen identisch und $\lfloor n/2 \rfloor$ paarweise unterschiedlich sind.
output: Das identische Element.

```
1 while true do
2 |   randomly choose i ∈ {1, 2, ..., n}
3 |   randomly choose j ∈ {1, 2, ..., n} \ {i}
4 |   if A[i] = A[j] then
5 |   |   return A[i]
```

(c) Was sind die Vor- und Nachteile beider Algorithmen?

Aufgabe 4: Algorithmen und Zufall

Bestimmen Sie den Erwartungswert der Ausgabe von Algorithmus 8.

Algorithmus 8:

```
1 s = 0
2 for i = 1 to n do
3 |   /* Random(a, b) gibt zufällig und gleichverteilt einen Wert zwischen a und b aus.
  |      */
4 |   if Random(1, 2 · i) ≤ i then
5 |   |   s = s + i
6 return s
```

Aufgabe 5: Hashing ausführen

Gegeben seien die folgenden Schlüssel k mit entsprechenden Werten der Hashfunktion $h(k)$:

k	N	I	C	E	A	L	G	O
$h(k)$	6	3	1	3	3	1	7	2

(a) Fügen Sie die Schlüssel in obiger Reihenfolge in eine Hash-Tabelle der Größe 8 ein. Kollisionen sollen durch Verkettung aufgelöst werden.

(b) Fügen Sie nun die Werte in der gleichen Reihenfolge in eine neue Hashtabelle mit Größe 8 ein. Lösen Sie Kollisionen mit linearem Sondieren auf. Geben Sie für jeden Schlüssel an, wie viele Felder der Tabelle betrachtet wurden.

(c) Fügen Sie wieder die Schlüssel in eine Tabelle der Größe 8 ein. Verwenden Sie dafür folgende Hashfunktion $l(k, i)$, welche Kollisionen mittels doppelten Hashing auflöst. Wobei m die Tabellengröße ist.

$$l(k, i) = (h(k) + i \cdot h_1(k)) \mod m$$

Im Folgenden sind die Werte von $h_1(k)$ für die Schlüssel angegeben. Geben Sie für jeden Schlüssel an, wie viele Felder der Tabelle betrachtet wurden.

k	N	I	C	E	A	L	G	O
$h_1(k)$	3	5	3	7	5	1	7	3

Aufgabe 6: Kosinus rekursiv

Es seien die folgenden Theoreme für $x, y \in \mathbb{R}$ gegeben:

Additionstheorem: $\cos(x + y) = 2 \cos x \cos y - \cos(x - y)$

Periodizität: $\cos(x + 2\pi) = \cos x$

Wir betrachten im Folgenden nur positive Winkel. Die Kosinusfunktion lässt sich für kleine Winkel x durch $\cos x \approx 1 - x^2/2$ annähern. Diese Näherung gelte für $x < 0,001$. Die Operatoren $+, -, \cdot, /$ und mod werden in konstanter Zeit ausgewertet.

(a) Geben Sie einen rekursiven Algorithmus zur Berechnung des Kosinus in Pseudocode an, der $\cos x$ für $x > 0$ mithilfe des Additionstheorems berechnet. Die gegebene Näherung soll als Abbruchkriterium der Rekursionsgleichung dienen. Die Laufzeit des Algorithmus soll in $\Theta(\log x)$ liegen.

(b) Nun soll die Worst-Case-Laufzeit durch Periodizität verbessert werden. Wie kann dies geschehen? Welche Laufzeit ergibt sich dann?

(c) Was passiert, wenn Sie für $x < 0,001$ die Näherung $\cos x \approx 1$ verwenden?

Aufgabe 7: 3-MergeSort

Gegeben sei folgende Algorithmus 10, das eine Variante vom bekannten MergeSort ist.

Algorithmus 10: `3MergeSort`(**int**[] A, **int** $l = 1$, **int** $r = A.length$)

```
1 if l < r then
2     m_1 = ⌊(r+l)/3⌋
3     m_2 = ⌊2(r+l)/3⌋
4     3MergeSort(A, l, m_1)
5     3MergeSort(A, m_1 + 1, m_2)
6     3MergeSort(A, m_2 + 1, r)
7     Merge(A, l, m_1, m_2)
8     Merge(A, l, m_2, r)
```

(a) Formulieren Sie die Laufzeit $T(n)$ von `3MergeSort` als Rekursionsgleichung in Abhängigkeit von der Länge des Feldes A.

(b) Bestimmen Sie die asymptotische Laufzeit von `3MergeSort` mithilfe der Meistermethode. Geben Sie dabei den Fall an.

Aufgabe 8: InsertionSort ∪ MergeSort

Obwohl die asymptotische worst-case Laufzeit von MergeSort $\Theta(n \log n)$ ist und die asymptotische worst-case Laufzeit von InsertionSort $\Theta(n^2)$ ist, läuft InsertionSort wegen der versteckten Konstanten in der O-Notation für kleine Eingaben oft schneller. Betrachte folgende Modifikation von MergeSort: Wir benutzen MergeSort bis wir n/k Teilfelder der Größe k haben. Diese sortieren wir mit InsertionSort und mergen sie anschließend mit dem bekannten Merge-Mechanismus von MergeSort.

(a) Zeigen Sie, dass InsertionSort die n/k Teilfelder insgesamt in $\Theta(nk)$ worst-case sortieren kann!

(b) Zeigen Sie, dass die oben beschriebene Variante von MergeSort in $\Theta(nk + n \log(n/k))$ worst-case läuft!

(c) Was ist der größte Wert von k als Funktion von n für die der modifizierte MergeSort Algorithmus die gleiche asymptotische worst-case Laufzeit hat wie die ursprüngliche Variante? (mit O-Notation)

(d) Wie sollte k in der Praxis gewählt werden?

Aufgabe 9: Paranoia

Gegeben sei folgender Algorithmus 11. Dabei ist das Set A eine endliche Menge von ganzzahligen Elementen der Länge n. Sie dürfen davon ausgehen, dass Einfüge-, Lösch- und Suchoperationen in $\mathcal{O}(1)$ möglich sind.

Algorithmus 11:

```
1  ParanoidMaximum(int[] A, ℓ = 1)
2      if A.length = ℓ then
3          return A[ℓ]
4      swap(A, ℓ, Random(ℓ, n))
5      a = A[ℓ]
6      b = ParanoidMaximum(A, ℓ + 1)
7      if b > a then
8          return b
9      else
10         for i = ℓ + 1 to A.length do
11             if A[i] > a then
12                 a = A[i]
13         return a
```

(a) Was berechnet Algorithmus 11 und ist die for-Schleife in der else Bedingung in den Zeilen 8-11 notwendig?

(b) Berechnen Sie die worst-case Laufzeit des Algorithmus, indem Sie eine Rekursionsgleichung aufstellen und diese mittels Substitionsmethode beweisen!
Hinweis: Hier eignet es sich in der Rekursionsgleichung die Anzahl der Vergleiche zu zählen.

Wir nehmen nun im Folgenden an, dass A paarweise verschiedene Zahlen enthält. Weiterhin ist die Anordnung der Zahlen in A zufällig gleichverteilt.

(c) Definieren Sie eine geeignete Zufallsvariable, die beide Fälle in den Zeilen 6-11 in Abhängigkeit der Rekursionstiefe i angibt!

(d) Geben Sie eine Formel für die erwartete Laufzeit mit Hilfe der definierten Zufallsvariable aus Teilaufgabe (c) an!

(e) Mit welcher Wahrscheinlichkeit tritt der Fall in den Zeilen 8-11 auf? Was ist der Erwartungswert Ihrer Zufallsvariable aus Teilaufgabe (c)?

(f) Berechnen Sie nun die erwartete Laufzeit mit Hilfe der Formel in Teilaufgabe (d) und der Wahrscheinlichkeit aus Teilaufgabe (e).

Aufgabe 10: Wahrscheinlichkeit und Erwartungswert

Auf dem Frühjahrs-Volksfest gibt es ein faires Glücksrad, welches mit den fünf Farben rot, grün, blau, gelb und glitzer angemalt ist. Der rote und der grüne Sektor nehmen jeweils 25 % der Fläche des Glücksrads ein, die anderen drei Farben nehmen jeweils ein Sechstel des Glücksrads ein.

(a) Hängt die Wahrscheinlichkeit, dass der Zeiger am Ende einer Drehung auf eine bestimmte Farbe zeigt, davon ab, ob die Farben zusammenhängen oder das Glücksrad in nach außen gehenden Streifen bemalt ist?

(b) Sie bekommen fünf Freifahrten, falls Sie „glitzer“ gedreht haben (Hauptpreis), drei Freifahrten, wenn Sie „gelb“ gedreht haben. Zwei Freifahrten gibt es, wenn der Zeiger „blau“ zeigt und eine Freifahrt ist der Gewinn bei „grün“. Die Farbe „rot“ geht leer aus. Wie viele Freifahrten gibt der Standbesitzer im Durchschnitt pro Besucher aus, wenn jeder Besucher einmal drehen darf?

(c) Die letzten neun Drehungen hatten alle das Ergebnis „glitzer“. Wie hoch ist die Wahrscheinlichkeit, dass die nächste, zehnte Drehung, erneut das Ergebnis „glitzer“ hat?

(d) Dem Standbetreiber ist es wichtig, dass die Kunden in der Warteschlange sehen, dass der Hauptpreis gewinnbar ist. Es sieht jeweils der nächste Kunde der Warteschlange und der vorherige Kunde, der gerade sein Gewinn abholt, das Ergebnis der aktuellen Drehung. Wenn an einem Tag n Kunden das Glücksrad drehen, wie viele sind dann im Durchschnitt Zeuge eines Hauptgewinns eines anderen Kunden?

Aufgabe 11: Zufall und Zufallsvariablen mit Würfeln

Gegeben seien zwei 6-seitige faire Würfel.

(a) Geben Sie den Ereignisraum Ω und dessen Kardinalität an, wenn beide Würfel nacheinander einmal geworfen werden.

(b) Mit welcher Wahrscheinlichkeit würfelt der erste Würfel eine bestimmte Augenzahl i und der zweite eine bestimmte Augenzahl j?

(c) Mit welcher Wahrscheinlichkeit haben beide Würfel die gleiche Augenzahl?

(d) Was ist der Erwartungswert des Maximums der beiden Augenzahlen eines Wurfes mit diesen zwei Würfeln?

Aufgabe 12: Leichte Klausuren

Jede ADS-Klausur wird von den Studierenden nach Ihrer Schwierigkeit bewertet. Die erste Klausur wurde im Semster 0 geschrieben, die letzte Klausur im Semester $n+1$. Jede Klausur hat einen eindeutigen Rang erhalten, sodass sich Bewertungen von 0 bis $n+1$ ergeben. Wir betrachten eine Klausur im Semester i als *leicht*, falls für deren Bewertung b_i gilt, dass $b_i < b_{i-1}$ und $b_i < b_{i+1}$ ist.

Wir nehmen an, dass jede mögliche Folge der Bewertungen gleich wahrscheinlich ist, also alle Permutationen von $(b_0, b_1, \ldots, b_n, b_{n+1})$ mit gleicher Wahrscheinlichkeit auftreten können. Als Studierender möchten Sie berechnen, wie viele leichte Klausuren Sie in n Semestern erwarten können.

(a) Sei K eine Zufallsvariable, die die Anzahl der leichten Klausuren in den Semestern 1 bis n beschreibt. Angenommen, Sie hätten eine Formel, die die Wahrscheinlichkeit $P(K = k)$ für $0 \leq k \leq n$ berechnet. Wie könnten Sie dann $E[K]$ für ein festes n ausrechnen?

(b) Im Allgemeinen ist es schwierig, eine Formel für $P(K = k)$ zu finden. Die Fragestellung können Sie aber trotzdem lösen, indem Sie mit Indikatorzufallsvariablen arbeiten. Definieren Sie eine Indikatorzufallsvariable K_i die einem Semester $0 < i < n+1$ den Wert 0 oder 1 zuweist, je nachdem, ob die Klausur einfach war oder nicht. Wir können Sie K in Abhängigkeit von K_i für $0 < i < n+1$ berechnen?

(c) Geben Sie außerdem eine Formel an, mit der Sie $E[K]$ in Abhängigkeit von $E[K_i]$ für $0 < i < n+1$ berechnen können. Welche Gesetzmäßigkeit des Erwartungswerts machen Sie sich zu Nutze?

(d) Nehmen Sie an, Sie kennen eine Formel, die $P(K_i = 1)$ für ein festes i berechnet. Geben Sie eine Formel an, mit der Sie $E[K_i]$ berechnen können. Formulieren Sie in Worten, was die Wahrscheinlichkeit $P(K_i = 1)$ ausdrückt.

(e) Wie groß ist die Wahrscheinlichkeit, dass die Bewertung b_i des Semesters i einen bestimmten Wert j aufweist *und* diese Klausur in Semester i eine leichte ist?

(f) Wie groß ist die Wahrscheinlichkeit $P(K_i = 1)$ für ein gegebenes i?

(g) Was ist die erwartete Anzahl von Klausuren in den Semestern 1 bis n, die als leicht empfunden werden?

Aufgabe 13: Indikatorzufallsvariablen und Maximum

Sei A ein Array paarweise unterschiedlicher ganzzahliger Zahlen der Länge n. Algorithmus 12 ermittelt die größte Zahl in A.

Algorithmus 12:

```
1 m = A[1]
2 for i = 2 to n do
3 |   if A[i] > m then
4 |   |   m = A[i]
5 |   return m
```

(a) Wie häufig wird im worst-case Zeile 4 ausgeführt?

(b) Definieren Sie eine Zufallsvariable X, die die Anzahl der Aktualisierungen von m angibt und definieren Sie zusätzlich Indikatorzufallsvariablen, mit der X gezählt werden kann.

(c) Berechnen Sie den Erwartungswert von X!

Aufgabe 14: Doppeltes Hashing

Welche der folgenden Funktionen eignen sich für eine Hashtabelle der Länge 25, wenn doppeltes Hashing verwendet wird und die Hashfunktion $h(k,i) = (h_0(k) + ih_1(k)) \bmod 25$ mit $h_0(k) = (4k+2) \bmod 25$ ist? Begründen Sie Ihre Entscheidungen.

(a) $h_1(k) = 1$

(b) $h_1(k) = 9 - (k \bmod 4)$

(c) $h_1(k) = k \bmod 17$

(d) $h_1(k) = (3 + 5k) \bmod 25$

(e) $h_1(k) = (4k - 1) \bmod 13$

Aufgabe 15: Gute Sondierfolgen erkennen

Gegeben sind folgende Hashfunktionen $h(k,i)$. Geben Sie an, um welche Methode der Kollisionsauflösung es sich handelt und gegebenfalls Probleme der Sondierfolgen, bei den gegebenen Tabellengrößen, an.

(a) $h(k,i) = (h_0(k) + (i+1) \cdot i) \mod m$, $m = 512$

(b) $h(k,i) = (h_0(k) + 2i) \mod m$, $m = 511$

(c) $h(k,i) = (h_0(k) + i \cdot h_1(k)) \mod m$, $h_1(k) = m - (2k \mod m) - 1$, $m = 16$.