Algorithmen und Datenstrukturen SoSe25

-Assignment 6-

Moritz Ruge
Matrikelnummer: 5600961
Lennard Wittenberg
Matrikelnummer: —

Juni 2025

Problem 1: Elementare Wahrscheinlichkeitsrechnung

Auf einem Tisch stehen N Kisten. In diese Kisten werden nacheinander unabhängig voneinander n Bälle geworfen, wobei jede Kiste mit gleicher Wahrscheinlichkeit getroffen wird.

a) Berechnen Sie die Wahrscheinlichkeit, dass Kiste i leer ist. Sei Yi die Zufalls- variable, die den Wert 1 annimmt, falls Kiste i leer ist, und 0 sonst. Geben Sie auch den Erwartungswert E[Yi] an. Hinweis: Wenn Kiste i leer bleibt, dann landen alle n Bälle in den $N-1$ anderen Kisten.

b) Sei X die Zufallsvariable, welche die Anzahl von leeren Kisten angibt. Berechnen Sie den Erwartungswert von X mit Hilfe der Erwartungswerte $E[Yi]$.

c) Bestimmen Sie, wie viele Bälle man benötigt, damit gilt: mit Wahrscheinlichkeit mindestens 1/2 enthält mindestens eine Kiste mindestens zwei Bälle.
$Hinweis$: Stellen Sie sich vor, die n Bälle werden nacheinander in die Kisten geworfen. Was ist die Wahrscheinlichkeit, dass der nächste Ball in einer leeren Kiste landet, wenn alle vorherigen Bälle in einer leeren Kiste gelandet sind? Verwenden Sie die ungemein nützliche Abschätzung $1+x \leq e^x$, welche für alle $x \in R$ gilt.

d) Professor Pinocchio hat eine Idee, um Hashtabellen zu vereinfachen. Wenn wir die Zahl N der Plätze in der Hashtabelle im Verhältnis zur Anzahl der zu speichernden Einträge n groß genug wählen, sollte die Wahrscheinlichkeit, dass Kollisionen auftreten, verschwindend gering werden (unter der Annahme, dass sich die Hashfunktion wie eine zufällige Funktion verhält). Dann könnte man auf die Kollisionsbehandlung verzichten. In Anbetracht von (c), was halten Sie von dem Vorschlag? (Wenn Sie Teil (c) nicht gelöst haben, dann bearbeiten Sie diesen Teil unter der Annahme, dass für $N^{1/3}$ Bälle mindestens eine Kollision auftritt.

Problem 2: Hashing: Worst-case-Analyse

Sei A eine Hashtabelle der Größe N , und $n \in N$ beliebig. Zeigen Sie: Für jede Schlüsselmenge K mit $|K| \geq (n-1)N+1$ und jede Hashfunktion $h : K \rightarrow \{0, \dots, N-1\}$ existiert eine Menge $S \subseteq Kmit|S| = n$, so dass alle Elemente von S auf denselben Eintrag in A abgebildet werden. Was bedeutet das für die worst-case Laufzeit von Hashing mit Verkettung? Wie verträgt sich das mit der Analyse aus der Vorlesung?

Gegeben:

- Hashtabelle A mit Größe N und $n \in \mathbb{N}$ beliebig
- Schlüsselmenge K mit $|K| \geq (n-1)N+1$
- Hashfunktion $h : K \rightarrow \{0, \dots, N-1\}$

Gesucht:

- Menge $S \subseteq K$ mit $|S| = n$ | so dass alle Elemnte von S auf denselben Eintrag in A abgebildet werden.
- Was bedeutet das für die worst-case Laufzeit von Hashing mit Verkettung?
- Wie verträgt sich das mti der Analyse aus der Vorlesung?

Lösung:

- $|K| \leq (n-1) * N + 1)$ Schlüssel
- Hashtabelle A der Größe N

$\Rightarrow$ Schubfachprinzip:
Kugeln: Schlüssel $((n-1)N+1)$
Fächer: Hashtabelle (N)
Zuordnung: Schlüssel durch die Hashfunktion in die jeweilige Hashtabelle einordnen

$\rightarrow$ Wir wollen die Teilmenge $S \subseteq K$ von Größe n, die alle denselben Hashwert haben $\rightarrow$ also im selben Fach landen.
$\rightarrow$ wir wollen höchsten n Schlüssel im selben Fach haben
$\Rightarrow$ wir wollen also maximal $n-1$ Schlüssel in jedem Fach, außer einem mit n Schlüsseln haben:

$\rightarrow$ d.h. wenn wir wissen möchten wie viel Schlüssel wir pro Slot(Fach) haben können wir das so ausdrücken: $n-1$

$\Rightarrow$ Wenn wir wissen möchten wie viel Schlüssel wir also maximal haben: $(n-1)*N$ | $*N$ anzahl der Fächer
$\rightarrow$ damit kriegen wir für jedes Fach maximal $n-1$ Schlüssel!

Nach der Aufgabenstellung haben wir aber mehr Gegeben!:

$|K| \geq (n-1) * N + 1$

$\Rightarrow$ Demnach haben wir genau ein Schlüssel mehr als wir brauchen und somit auch ein Fach was mehr als $n-1$ Schlüssel hat und genau n Schlüssel. Dies bildet die gesuchte Menge $S \subseteq K$, mit $|S| = n$ und $\forall x \in S : h(x) = i$ für irgendein $i \in \{0, \ldots, N-1\}$

$\square$

Beispiel:

- Fächer N: 5
- Schlüssel $n = 8$
- $\Rightarrow (n-1) * N + 1 = (8-1) * 5 + 1 = 36$
- 36 auf 5 Fächer sind:
  - 4 Fächer mit 7 Schlüsseln &
  - 1 Fach mit 8 Schlüsseln

Was bedeutet das für die worst-case Laufzeit von Hashing mit Verkettung?

- Bei Hashing mit Verkettung speichern wir alle Schlüssel, die denselben Hashwert haben, in einer Liste in der jeweiligen Stelle der Hashtabelle
- d.h. im worst-case kann es passieren, dass alle Schlüssel auf denselben Slot gespeichert werden (z.B. bei schlechter Hashfunktion)
- Im Schlimmsten Fall muss man also eine Liste mit n Elementen durchsuchen

Worst-Case-Laufzeit für Hashing mit Verkettung ist also:
$\Rightarrow O(n)$

Wie verträgt sich das mit der Analyse aus der Vorlesung?

- In der Vorlesung wird für Hashing mit Verkettung eine Laufzeit von $O(1 + \frac{n}{m})$ angegeben
- $\frac{n}{m}$ ist hierbei der Ladefaktor n: Schlüssel, m: die Größe der Hashtabelle(Slots)
- Im besten Falle sollte der Ladefaktor zwischen 1 & 3 sein. Hierbei handelt es sich also um die Praktische Anwendung von Hashing mit Verkettung oder auch Average-Case-Laufzeit
- Bei gute Wahl vom Ladefaktor kann also die Laufzeit $O(1)$ werden, wenn der Ladefaktor zu ≈ 1 wird
- Im Vergleich haben wir also den Worst-Case von $O(n)$ und den Average-Case von $O(1)$

Problem 3: Hashing im Selbstversuch

Fügen Sie nacheinander die Schlüssel 5, 28, 19, 15, 20, 33, 12, 17, 10 in eine Hashtabelle der Größe 9 ein. Die Hashfunktion sei $h(k) = kmod9$. Sie mit Die Konflikte werden mit Verkettung gelöst. Illustrieren Sie jeweils die einzelnen Schritte. Wiederholen Sie dann mit der Hashfunktion $h(k) = kmod6$ und einer Hashtabelle der Größe 6.