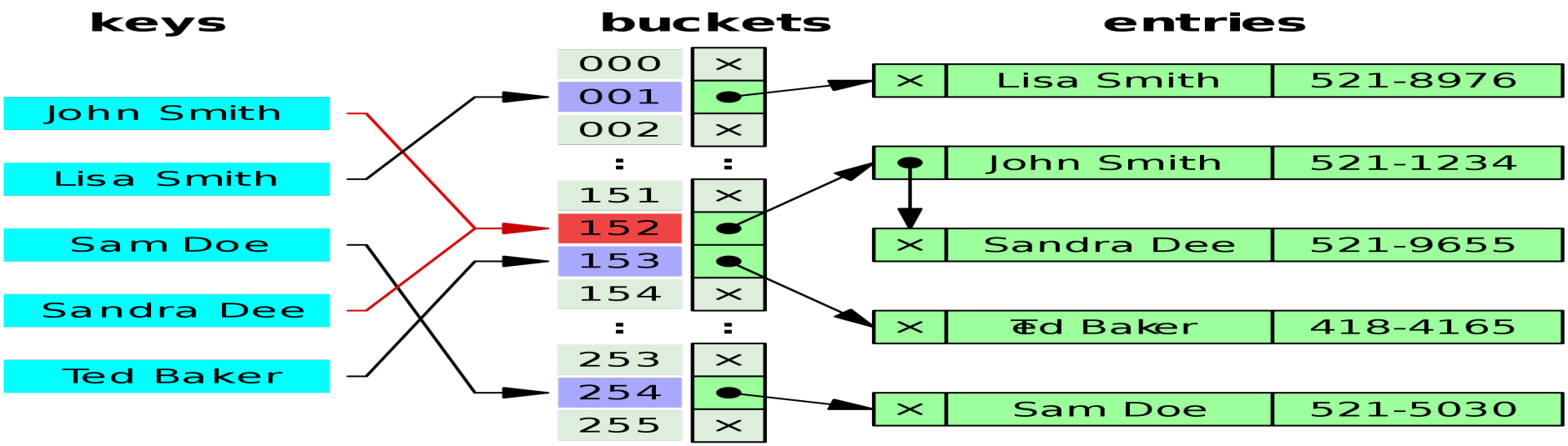
Karlsruher Institut für Technologie

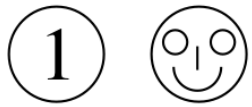
Algorithmen I - Tutorium 4

Sebastian Schmidt – *isibboi@gmail.com*

Arbeitsgruppe Kryptographie und Sicherheit



Wahrscheinlichkeitstheorie



Elementarereignisse Ω

Ereignisse: Teilmengen von Ω

p_x =Wahrscheinlichkeit von $x \in \Omega$. $\sum_x p_x = 1$!

Gleichverteilung: $p_x = \frac{1}{|\Omega|}$

$\mathbb{P}[\mathscr{E}] = \sum_{x \in \mathscr{E}} p_x$

Zufallsvariable (ZV) $X_0 : \Omega \to \mathbb{R}$

0-1-Zufallsvariable (Indikator-ZV) $I : \Omega \to \{0,1\}$

Erwartungswert $\mathrm{E}[X] = \sum_{y \in \Omega} p_y X(y)$

Linearität des Erwartungswerts: $\mathrm{E}[X+Y] = \mathrm{E}[X] + \mathrm{E}[Y]$

Hash-Beispiel

Hash-Funktionen $\{0..m-1\}^{\mathrm{Key}}$

$\mathscr{E}_{42} = \{h \in \Omega : h(4) = h(2)\}$

$p_h = m^{-|\mathrm{Key}|}$

$\mathbb{P}[\mathscr{E}_{42}] = \frac{1}{m}$

$X = |\{e \in M : h(e) = 0\}|$

$\mathrm{E}[X] = \frac{|M|}{m}$

Hashing mit einfach verketteten Listen

- Hashtabelle mit $h_n(x) := x \mod n$
  - Wie groß sollte die Hashtabelle sein?
  - Gegeben $\{36, 78, 50, 1, 92, 15, 43, 99, 64\}$. Füge diese Zahlen in eine Hashtabelle mit Hashfunktion h_5 und h_7 ein.
- Gebt Beispiele für gute und schlechte Hashfunktionen
- Laufzeiten: `insert`, `find` und `remove`

Hashing mit einfach verketteten Listen

- Hashtabelle mit $h_n(x) := x \mod n$
  - Wie groß sollte die Hashtabelle sein?
  - Gegeben $\{36, 78, 50, 1, 92, 15, 43, 99, 64\}$. Füge diese Zahlen in eine Hashtabelle mit Hashfunktion h_5 und h_7 ein.
- Gebt Beispiele für gute und schlechte Hashfunktionen
- Laufzeiten: `insert`, `find` und `remove`

Hashing mit einfach verketteten Listen

- Hashtabelle mit $h_n(x) := x \mod n$
  - Wie groß sollte die Hashtabelle sein?
  - Gegeben $\{36, 78, 50, 1, 92, 15, 43, 99, 64\}$. Füge diese Zahlen in eine Hashtabelle mit Hashfunktion h_5 und h_7 ein.
- Gebt Beispiele für gute und schlechte Hashfunktionen
- Laufzeiten: `insert`, `find` und `remove`

Hashing mit einfach verketteten Listen

- Hashtabelle mit $h_n(x) := x \mod n$
  - Wie groß sollte die Hashtabelle sein?
  - Gegeben $\{36, 78, 50, 1, 92, 15, 43, 99, 64\}$. Füge diese Zahlen in eine Hashtabelle mit Hashfunktion h_5 und h_7 ein.
- Gebt Beispiele für gute und schlechte Hashfunktionen
- Laufzeiten: `insert`, `find` und `remove`

Hashing mit einfach verketteten Listen

- Hashtabelle mit $h_n(x) := x \mod n$
  - Wie groß sollte die Hashtabelle sein?
  - Gegeben $\{36, 78, 50, 1, 92, 15, 43, 99, 64\}$. Füge diese Zahlen in eine Hashtabelle mit Hashfunktion h_5 und h_7 ein.
- Gebt Beispiele für gute und schlechte Hashfunktionen
- Laufzeiten: `insert`, `find` und `remove`

Anwendungsbeispiele für Hashtabellen

Hashtabellen sind besser als Bäume, wenn man die erwartete Laufzeit betrachtet.

- Fallen euch konkrete Beispiele oder Gegenbeispiele ein?

Anwendungsbeispiele für Hashtabellen

Hashtabellen sind besser als Bäume, wenn man die erwartete Laufzeit betrachtet.

- Fallen euch konkrete Beispiele oder Gegenbeispiele ein?

Beweis über Worst-Case

Seien $m, n \in \mathbb{N}$. n ist die Anzahl der Elemente, die in eine Hashtabelle der Größe m eingefügt werden.
Sei U ein Universum mit $|U| \geq mn$.
Zeige, dass eine Teilmenge $M \subset U$ existiert mit $|M| \geq n$, sodass alle Elemente aus M dem selben Slot der Hashtabelle zugeordnet werden.

Verbesserung des Worst-Case

„Nach dem dritten Glas Bier behauptet ein Kommilitone, man könne Hashing mit verketteten Listen entscheidend verbessern, indem man die verketteten Listen stets sortiert halte.“

- Wie ändert sich dadurch die Worst-Case Laufzeit von `insert`, `find` und `remove`?
- Was muss man tun, um die Sortiertheit auszunutzen?
- Wie ändert sich nun die Laufzeit?
- Kann man durch Amortisierung noch was verbessern?

Verbesserung des Worst-Case

„Nach dem dritten Glas Bier behauptet ein Kommilitone, man könne Hashing mit verketteten Listen entscheidend verbessern, indem man die verketteten Listen stets sortiert halte.“

- Wie ändert sich dadurch die Worst-Case Laufzeit von `insert`, `find` und `remove`?
- Was muss man tun, um die Sortiertheit auszunutzen?
- Wie ändert sich nun die Laufzeit?
- Kann man durch Amortisierung noch was verbessern?

Verbesserung des Worst-Case

„Nach dem dritten Glas Bier behauptet ein Kommilitone, man könne Hashing mit verketteten Listen entscheidend verbessern, indem man die verketteten Listen stets sortiert halte.“

- Wie ändert sich dadurch die Worst-Case Laufzeit von `insert`, `find` und `remove`?
- Was muss man tun, um die Sortiertheit auszunutzen?
- Wie ändert sich nun die Laufzeit?
- Kann man durch Amortisierung noch was verbessern?

Verbesserung des Worst-Case

„Nach dem dritten Glas Bier behauptet ein Kommilitone, man könne Hashing mit verketteten Listen entscheidend verbessern, indem man die verketteten Listen stets sortiert halte.“

- Wie ändert sich dadurch die Worst-Case Laufzeit von `insert`, `find` und `remove`?
- Was muss man tun, um die Sortiertheit auszunutzen?
- Wie ändert sich nun die Laufzeit?
- Kann man durch Amortisierung noch was verbessern?