TECHNISCHE UNIVERSITÄT DARMSTADT
FACHGEBIET THEORETISCHE INFORMATIK

PROF. JOHANNES BUCHMANN
NABIL ALKEILANI ALKADRI
NINA BINDEL
PATRICK STRUCK

Algorithmen und Datenstrukturen

SoSe 2018

TECHNISCHE UNIVERSITÄT DARMSTADT

5. Lösungsblatt — 14.05.2018
v1.1

P1 Hashfunktionen

Bestimmen Sie, inwiefern sich folgende Funktionen für den Einsatz als Hashfunktion einer Hashtabelle der Größe m eignen. Nehmen Sie als Schlüssel alle natürlichen Zahlen an. Begründen Sie ihre Antwort.

1. $\lfloor \frac{8}{x} \rfloor \ mod\ 10, m = 10$
2. $x\ mod\ 25, m = 26$
3. $x\ mod\ 25, m = 25$
4. Die führenden 6 Bit vom Secure Hash Algorithm(SHA), $m = 64$
   https://de.wikipedia.org/wiki/Secure_Hash_Algorithm

Lösung.

1. Die Funktion ist einfach zu berechnen, jedoch nicht surjektiv, d.h. dass alle Werte im Zielraum getroffen werden. Sie eignet sich daher nicht.
2. Auch diese Funktion ist nicht surjektiv und daher nicht geeignet.
3. Diese Funktion lässt sich einfach berechnen, ist surjektiv und verteilt die Werte gleichmäßig im Zielraum. Sie ist grundsätzlich als Hashfunktion geeignet. Es gibt jedoch auch Funktionen, die die Werte besser streuen, sodass benachbarte Schlüssel im Zielraum nicht mehr benachbart sind.
4. Bei SHA handelt es sich um ein Hashverfahren aus der Kryptographie. Es ist sehr schwer Kollisionen zu finden. Es ist anzunehmen, dass das Verfahren surjektiv ist. Außerdem streut es sehr gut. In der Praxis würde man das Verfahren jedoch wahrscheinlich nicht als Hashfunktion einer Hashtabelle einsetzen, da es im Vergleich langsam ist und eine Hashtabelle in erster Linie Zugriffe beschleunigen soll.

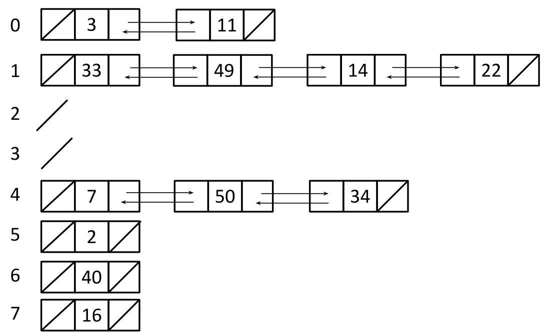
P2 Hashtabelle - Verkettung

Fügen Sie folgende Schlüssel in der gleichen Reihenfolge in eine Hashtabelle mit $m = 8$ Plätzen ein: 16, 34, 50, 7, 22, 14, 49, 33, 40, 11, 3, 2. Sollte es zu Kollisionen kommen, benutzen Sie doppelt verkettete Listen, um diese aufzulösen. Verwenden Sie dabei die Hashfunktion $h(k) = \lfloor 8 \cdot (0,37 \cdot k\ mod\ 1) \rfloor$.

Lösung. Die Hashwerte der Schlüssel sind:

$$
\begin{array}{rrrr}
h(16) = 7, & h(34) = 4, & h(50) = 4, & h(7) = 4, \\
h(22) = 1, & h(14) = 1, & h(49) = 1, & h(33) = 1, \\
h(40) = 6, & h(11) = 0, & h(3) = 0, & h(2) = 5.
\end{array}
$$

Schlüssel werden immer am vorderen Ende der verketteten Liste eingefügt. Nach dem Einfügen aller Schlüssel sieht die Hashtabelle wie folgt aus:



P3 Kollisionen

Gegeben sei eine Hashfunktion H, die n verschiedene Schlüssel k_i auf m verschiedene Ausgabewerte h_i abbildet. Nehmen Sie an, dass H jeden Schlüssel k mit der Wahrscheinlichkeit $1/m$ auf jeden der Ausgabewerte abbildet, d.h.

$$\Pr[\mathsf{H}(k_i) = h_j] = \frac{1}{m} \quad \forall i, j.$$

Geben Sie den Erwartungswert für die Anzahl der Kollisionen an, d.h. bestimmen Sie

$$E[|K|] = E[|\{\{r, s\} : r \neq s \text{ und } \mathsf{H}(k_r) = \mathsf{H}(k_s)\}|].$$

Lösung. Sei k_r ein Schlüssel mit Hashwert $\mathsf{H}(k_r)$. Die Wahrscheinlichkeit, dass für einen zweiten Schlüssel k_s $(s \neq r)$ $\mathsf{H}(k_s) = \mathsf{H}(k_r)$ gilt, ist nach unserer Annahme $\frac{1}{m}$. Ein Paar (r, s) liefert also mit Wahrscheinlichkeit $\frac{1}{m}$ eine Kollision der Hashfunktion.
Um die erwartete Zahl der Elemente der Menge K zu bestimmen, müssen wir diese Wahrscheinlichkeit noch mit der Anzahl der Paare (r, s) mit $1 \leq r < s \leq n$ multiplizieren. Diese ist durch den Binomialkoeffizient $\binom{n}{2}$ gegeben. Wir erhalten also

$$E[|K|] = \binom{n}{2} \cdot \frac{1}{m} = \frac{n \cdot (n-1)}{2} \cdot \frac{1}{m} = \frac{n \cdot (n-1)}{2 \cdot m}.$$

P4 Hashtabelle - Doppeltes Hashing

Fügen Sie die Schlüssel 27, 45, 4, 33, 17, 26, 10, 31, 35 in eine Hashtabelle der Größe $m = 12$ ein. Die Hashtabelle soll offene Adressierung benutzen und Konflikte mit doppelten Hashing auflösen. Verwenden Sie die Hashfunktion
$h(k, i) = (k \ mod \ m + i \cdot h'(k)) \ mod \ m$ mit $h'(k) = k \ mod \ 7$

Geben Sie die entstehende Tabelle und bestimmen Sie für jeden Einfügevorgang den finalen Wert von i.

Lösung. Beim ersten Einfügeversuch ist i immer gleich 0. Für jeden folgenden Versuch wird i um 1 erhöht.

$h(27, 0) = 3$ Platz 3 ist noch nicht belegt: $i = 0$

$h(45, 0) = 9$ Platz 9 ist noch nicht belegt: $i = 0$

$h(4, 0) = 4$ Platz 4 ist noch nicht belegt: $i = 0$

$h(33, 0) = 9$ Platz 9 ist bereits belegt: $i = i + 1 = 1$
$h(33, 1) = 2$ Platz 2 ist noch nicht belegt: $i = 1$

$h(17,0) = 5$ Platz 5 ist noch nicht belegt: $i = 0$

$h(26,0) = 2$ Platz 2 ist bereits belegt: $i = i + 1 = 1$
$h(26,1) = 7$ Platz 7 ist noch nicht belegt: $i = 1$

$h(10,0) = 10$ Platz 10 ist noch nicht belegt: $i = 0$

$h(31,0) = 7$ Platz 7 ist bereits belegt: $i = i + 1 = 1$
$h(31,1) = 10$ Platz 10 ist bereits belegt: $i = i + 1 = 2$
$h(31,2) = 1$ Platz 1 ist noch nicht belegt: $i = 2$

$h(35,0) = 11$ Platz 11 ist noch nicht belegt: $i = 0$

Am Ende ergibt sich folgende Tabelle:

Hashwert	0	1	2	3	4	5	6	7	8	9	10	11
Key		31	33	27	4	17		26		45	10	35

H1 Hashtabelle - Lineares Sondieren

Eine leere Hashtabelle mit $m = 9$ Plätzen löst Konflikte mittels linearem Sondieren. Fügen Sie nacheinander die Werte 22, 6, 30, 13, 27, 11, 25 ein. Die Hashfunktion lautet $h'(k) = k \ mod \ m$. Als Parameter c wurde 3 gewählt. Es gilt also $h(k) = (h'(k) + 3i) \ mod \ m$. Bestimmen Sie die resultierende Hashtabelle. Geben Sie außerdem für jeden Wert an bei welchen i das Einfügen erfolgreich war.

Lösung. Beim ersten Einfügeversuch ist i immer gleich 0. Für jeden folgenden Versuch wird i um 1 erhöht.

$h(22,0) = 4$ Platz 4 ist noch nicht belegt: $i = 0$

$h(6,0) = 6$ Platz 6 ist noch nicht belegt: $i = 0$

$h(30,0) = 3$ Platz 3 ist noch nicht belegt: $i = 0$

$h(13,0) = 4$ Platz 4 ist bereits belegt: $i = i + 1 = 1$
$h(13,1) = 7$ Platz 7 ist noch nicht belegt: $i = 1$

$h(27,0) = 0$ Platz 0 ist noch nicht belegt: $i = 0$

$h(11,0) = 2$ Platz 2 ist noch nicht belegt: $i = 0$

$h(25,0) = 7$ Platz 7 ist bereits belegt: $i = i + 1 = 1$
$h(25,1) = 1$ Platz 1 ist noch nicht belegt: $i = 1$

Dies entspricht folgender Tabelle:

Hashwert	0	1	2	3	4	5	6	7	8
Key	27	25	11	30	22	—	6	13	—

H2 Anzahl Einfügeversuche bei offener Adressierung

Zeigen Sie, dass der Erwartungswert der nötigen Einfügeversuche eines Schlüssels in eine Hashtabelle mit einem Belegungsgrad $\alpha = \frac{n}{m}$ durch $\frac{1}{1-\alpha}$ beschränkt ist. Dabei bezeichnet n die Anzahl der bereits eingefügten Elemente und m die verfügbaren Plätze. Sie können annehmen, dass eine Hashfunktion verwendet wurde, die die Schlüssel gleichmäßig

verteilt.
Hinweis: Es gilt

$$\frac{1}{1-\alpha} = 1 + \alpha + \alpha^2 + \ldots = \sum_{i=0}^{\infty} \alpha^i$$

und

$$E(X) = \sum_{i=1}^{\infty} P(X \geq i)$$

Lösung. Da wir von einer guten Hashfunktion ausgehen, bei der jeder Ablageort die gleiche Wahrscheinlichkeit hat, beträgt die Wahrscheinlichkeit, dass der erste Versuch scheitert $\frac{n}{m}$, der zweite $\frac{n-1}{m-1}$, usw. Also ergibt sich für die benötigten Versuche folgende Wahrscheinlichkeit:

$$P(X \geq i) = \frac{n}{m} \cdot \frac{n-1}{m-1} \cdot \frac{n-2}{m-2} \cdots \frac{n-i+2}{m-i+2} \leq \left(\frac{n}{m}\right)^{i-1} = \alpha^{i-1}$$

Mit dem zweiten Hinweis kann man nun den Erwartungswert berechnen:

$$E(X) \leq \sum_{i=1}^{\infty} \alpha^{i-1} = \sum_{i=0}^{\infty} \alpha^i = \frac{1}{1-\alpha} \quad \square$$