



# Algorithmtheorie

## 05 - Hashing

Prof. Dr. S. Albers  
Prof. Dr. Th. Ottmann

- Einführung
- Universelles Hashing
- Perfektes Hashing

# Das Wörterbuch-Problem

---

**Gegeben:** Universum  $U = [0 \dots N-1]$ , wobei  $N$  eine natürliche Zahl ist.

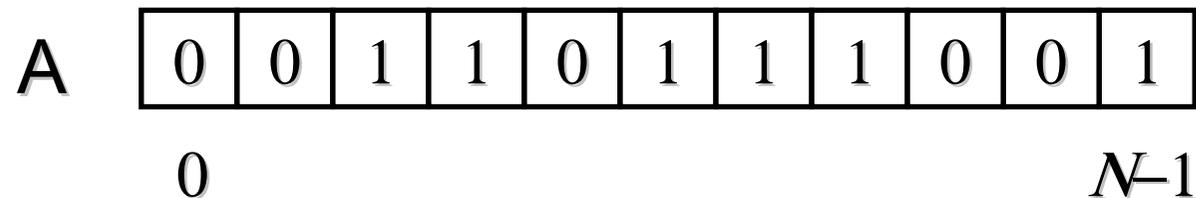
**Ziel:** Verwalte Menge  $S \subseteq U$  mit folgenden Operationen.

- **Suche( $x, S$ ):** Ist  $x \in S$ ?
- **Einfüge( $x, S$ ):** Füge  $x$  zu  $S$  hinzu, sofern noch nicht vorhanden.
- **Entferne( $x, S$ ):** Entferne  $x$  aus  $S$ .

# Triviale Implementierung

Array  $A[0 \dots N-1]$  wobei  $A[i] = 1 \Leftrightarrow i \in S$

Jede Operation hat Laufzeit  $O(1)$ , aber der Platzbedarf ist  $\Theta(N)$ .



**Ziel:** Platzbedarf  $O(|S|)$  und erwartete Laufzeit  $O(1)$ .

# Idee des Hashings

---

Verwende ein **Array** der Länge  $O(|S|)$ .

Berechne die **Position**, an der ein Element abgespeichert wird mit Hilfe einer **Funktion** aus dem **Schlüssel**.

Universum	$U = [0 \dots N-1]$
Hash-Tafel	Array $T[0 \dots m-1]$
Hash-Funktion	$h: U \rightarrow [0 \dots m-1]$

Ein Element  $x \in S$  wird in  $T[h(x)]$  gespeichert.

# Beispiel



$N = 100$ ;  $U = [0 \dots 99]$ ;  $m = 7$ ;  $h(x) = x \bmod 7$ ;  $S = \{3, 19, 22\}$

0	
1	22
2	
3	3
4	
5	19
6	

Soll als nächstes 17 eingefügt werden, so tritt eine **Kollision** auf, denn  $h(17) = 3$ .

# Lösungsmöglichkeiten für Kollisionen

---

- Hashing mit Verkettung:  $T[i]$  enthält eine **Liste** von Elementen.
- Hashing mit offener Adressierung: Statt einer **Adresse** für ein Element gibt es  **$m$  viele**, die der Reihe nach ausprobiert werden.
- Universelles Hashing: Wähle eine **Hash-Funktion**, so dass **wenige Kollisionen** entstehen. Kollisionen werden durch Verkettung aufgelöst.
- Perfektes Hashing: Wähle eine **Hash-Funktion**, so dass **keine Kollisionen** entstehen.

# Universelles Hashing



**Idee:** Verwende eine **Klasse  $H$**  von Hash-Funktionen. Die tatsächlich verwendete Hash-Funktion  $h \in H$  wird **zufällig** aus  $H$  gewählt.

**Ziel:** Für jedes  $S \subseteq U$  soll die erwartete Laufzeit jeder Operation  $O(1 + \beta)$  sein, wobei  $\beta = |S|/m$  der **Lastfaktor** der Tafel ist.

**Eigenschaft von  $H$ :** Für zwei beliebige Elemente  $x, y \in U$  führen nur wenige  $h \in H$  zu einer Kollision ( $h(x) = h(y)$ ).

# Universelles Hashing

**Definition:** Seien  $N$  und  $m$  natürliche Zahlen. Eine Klasse  $H \subseteq \{ h : [0 \dots N-1] \rightarrow [0 \dots m-1] \}$  heißt **universell**, wenn für alle  $x, y \in U = [0 \dots N-1]$ ,  $x \neq y$ , gilt:

$$\frac{|\{h \in H : h(x) = h(y)\}|}{|H|} \leq \frac{1}{m}$$

**Intuitiv:** Ein zufällig gewähltes  $h$  ist genau so gut, wie wenn die Tafelpositionen der Elemente zufällig gewählt würden.

# Eine universelle Klasse von Funktionen

Seien  $N, m$  natürliche Zahlen, wobei  $N$  prim ist.

Für Zahlen  $a \in \{1, \dots, N-1\}$  und  $b \in \{0, \dots, N-1\}$  sei

$h_{a,b} : U = [0 \dots N-1] \rightarrow \{0, \dots, m-1\}$  definiert durch:

$$h_{a,b}(x) = ((ax + b) \bmod N) \bmod m$$

**Satz:**  $H = \{h_{a,b}(x) \mid 1 \leq a < N \text{ und } 0 \leq b < N\}$  ist eine **universelle Klasse** von Hash-Funktionen.

Betrachte festes Paar  $x, y$  mit  $x \neq y$ .

$$h_{a,b}(x) = ((ax+b) \bmod N) \bmod m \quad h_{a,b}(y) = ((ay+b) \bmod N) \bmod m$$

1. Paare  $(q,r)$  mit  $q = (ax+b) \bmod N$  und  $r = (ay+b) \bmod N$  durchlaufen für variables  $a,b$  den **gesamten Bereich**  
 $0 \leq q,r < N$  mit  $q \neq r$

--  $q \neq r$ :  $q = r$  impliziert  $a(x-y) = cN$

-- Verschiedene Paare  $a,b$  ergeben verschiedene Paare  $(q,r)$ .

$$(ax+b) \bmod N = q \quad (ay+b) \bmod N = r$$

$$(a'x+b') \bmod N = q \quad (a'y+b') \bmod N = r$$

$$\text{implizieren } (a-a')(x-y) = cN$$

Festes Paar  $x, y$  mit  $x \neq y$  .

$$h_{a,b}(x) = ((ax+b) \bmod N) \bmod m \quad h_{a,b}(y) = ((ay+b) \bmod N) \bmod m$$

2. **Wieviele Paare  $(q,r)$**  mit  $q = (ax+b) \bmod N$  und  $r = (ay+b) \bmod N$  werden auf die **gleiche Restklasse mod  $m$**  abgebildet?

Für festes  $q$  gibt es nur  $(N-1)/m$  Zahlen  $r$ , mit  
 $q \bmod m = r \bmod m$  und  $q \neq r$ .

$$|\{h \in H : h(x) = h(y)\}| \leq N(N-1)/m = |H|/m$$

- Annahmen:
1.  $h$  wird zufällig (gemäß Gleichverteilung) aus einer universellen Klasse  $H$  gewählt.
  2. Kollisionen werden durch Verkettung gelöst.

Für  $h \in H$  und  $x, y \in U$  sei

$$\delta_h(x, y) = \begin{cases} 1 & h(x) = h(y) \text{ und } x \neq y \\ 0 & \text{sonst} \end{cases}$$

$\delta_h(x, S) = \sum_{y \in S} \delta_h(x, y)$  ist die Anzahl der von  $x$  verschiedenen

Elemente in  $\pi[h(x)]$ , wenn  $S$  gespeichert wird.

# Analyse der Operationen

**Satz:** Sei  $H$  eine universelle Klasse und  $S \subseteq U = [0 \dots N-1]$  mit  $|S| = n$ .

1. Für  $x \in U$  gilt:

$$\frac{1}{|H|} \sum_{h \in H} (1 + \delta_h(x, S)) \leq \begin{cases} 1 + n/m & x \notin S \\ 1 + (n-1)/m & x \in S \end{cases}$$

2. Die erwartete Laufzeit einer Suche-, Einfüge bzw. Lösche-Operation ist  $O(1 + \beta)$ , wobei  $\beta = n/m$  der Lastfaktor ist.

$$\begin{aligned} 1. \quad \sum_{h \in H} (1 + \delta_h(x, S)) &= |H| + \sum_{h \in H} \sum_{y \in S} \delta_h(x, y) \\ &= |H| + \sum_{y \in S} \sum_{h \in H} \delta_h(x, y) \\ &\leq |H| + \sum_{y \in S \setminus \{x\}} \frac{|H|}{m} \\ &\leq \begin{cases} |H| (1 + n/m) & x \notin S \\ |H| (1 + (n-1)/m) & x \in S \end{cases} \end{aligned}$$

2. Folgt aus 1.

# Wahrscheinlichkeit einer schlechten Wahl



**Satz:** Sei  $H$  eine universelle Klasse,  $S \subseteq U$  mit  $|S| = n$  und  $x \in U$ .

Sei

$$\mu = \frac{1}{|H|} \sum_{h \in H} \delta_h(x, S).$$

Dann gilt für alle  $c \geq 1$ :  $\text{Prob} [ \delta_h(x, S) \geq c \mu ] \leq 1/c$

**Beweis:** Die zu beweisende Aussage ist äquivalent zu:

$$\frac{1}{|H|} |\{h : \delta_h(x, S) \geq c\mu\}| \leq \frac{1}{c}$$

Definiere  $H_c = \{h : \delta_h(x, S) \geq c\mu\}$

Nach Definition:

$$\begin{aligned}\mu &= \frac{1}{|H|} \sum_{h \in H} \delta_h(x, S) \\ &\geq \frac{1}{|H|} \sum_{h \in H_c} \delta_h(x, S) \\ &\geq \frac{1}{|H|} \sum_{h \in H_c} c\mu \\ &= c\mu \frac{|H_c|}{|H|}\end{aligned}$$

Also  $|H_c| \leq |H|/c$ .

# Perfektes Hashing

---

Wähle eine **Hash-Funktion**, die für die abzuspeichernde Menge  $S$  **injektiv** ist.  $S$  sei im Voraus bekannt.

## Zweistufiges Hashverfahren

1. Die erste Stufe verteilt  $S$  auf “kurze Listen”.  
(Hashing mit Verkettung)
2. In der zweiten Stufe wird für jede Liste eine **eigene injektive Hash-Funktion** benutzt.

# Konstruktion von injektiven Hashfunktionen



Sei  $U = [0 \dots N-1]$

Für  $k \in \{1, \dots, N-1\}$  sei

$$\begin{aligned} h_k : U &\rightarrow \{0, \dots, m-1\} \\ x &\rightarrow ((kx) \bmod N) \bmod m \end{aligned}$$

Sei  $S \subseteq U$ . Kann  $k$  so gewählt werden, dass  $h_k$  eingeschränkt auf  $S$  injektiv ist?

$h_k$  eingeschränkt auf  $S$  ist injektiv, wenn für alle  $x, y \in S$ ,  $x \neq y$ , gilt

$$h_k(x) \neq h_k(y)$$

# Maß für Verletzung der Injektivität

Für  $0 \leq i \leq m-1$  und  $1 \leq k \leq N-1$  sei

$$b_{ik} = |\{x \in S : h_k(x) = i\}|$$

Dann gilt:

$$|\{(x,y) \in S^2 : x \neq y \text{ und } h_k(x) = h_k(y) = i\}| = b_{ik} (b_{ik} - 1)$$

Definiere

$$B_k = \sum_{i=0}^{m-1} b_{ik} (b_{ik} - 1)$$

$B_k$  misst, wie wenig injektiv  $h_k$  eingeschränkt auf  $S$  ist.

**Lemma 1:**  $h_k$  eingeschränkt auf  $S$  ist injektiv  $\Leftrightarrow B_k < 2$

**Beweis:**

$$\begin{aligned} B_k < 2 &\Rightarrow B_k \leq 1 \Rightarrow b_{ik}(b_{ik} - 1) \in \{0, 1\} \text{ für alle } i \\ &\Rightarrow b_{ik} \in \{0, 1\} \Rightarrow h_k \text{ eingeschränkt auf } S \text{ ist injektiv} \end{aligned}$$

$$\begin{aligned} h_k \text{ eingeschränkt auf } S \text{ ist injektiv} &\Rightarrow b_{ik} \in \{0, 1\} \text{ für alle } i \\ &\Rightarrow B_k = 0 \end{aligned}$$

**Lemma 2:** Sei  $N$  Primzahl,  $S \subseteq U = [0 \dots N-1]$  mit  $|S| = n$ . Dann gilt

$$\sum_{k=1}^{N-1} \underbrace{\sum_{i=0}^{m-1} b_{ik} (b_{ik} - 1)}_{B_k} \leq 2 \frac{n(n-1)}{m} (N-1)$$

**Bemerkung:**  $\sum_i b_{ik} (b_{ik} - 1)$  ist zweimal die Anzahl der Kollisionen, falls  $h_k$  als Hash-Funktion gewählt wird. Die mittlere Anzahl der Kollisionen ist somit  $n(n-1)/m$ .

Ist  $m > n(n-1)$ , so ist die mittlere Anzahl  $< 1$ ,

d.h. es existiert ein  $h_k$ , das eingeschränkt auf  $S$  injektiv ist.

## Beweis von Lemma 2

$$\begin{aligned} & \sum_{k=1}^{N-1} \sum_{i=0}^{m-1} b_{ik} (b_{ik} - 1) \\ &= \sum_{k=1}^{N-1} \sum_{i=0}^{m-1} |\{(x, y) \in S^2 : x \neq y, h_k(x) = h_k(y) = i\}| \\ &= \sum_{\substack{(x,y) \in S^2 \\ x \neq y}} |\{k : h_k(x) = h_k(y)\}| \end{aligned}$$

Sei  $(x, y) \in S^2$ ,  $x \neq y$ , fest. Wie viele  $k$  mit  $h_k(x) = h_k(y)$  gibt es?

## Beweis von Lemma 2

$$h_k(x) = h_k(y)$$

$$\Leftrightarrow ((kx) \bmod N) \bmod m = ((ky) \bmod N) \bmod m$$

$$\Leftrightarrow \underbrace{(kx \bmod N - ky \bmod N)}_{=:g(k)} \bmod m = 0$$

Es gilt  $-(N-1) \leq g(k) \leq (N-1)$  und  $g(k) = im$  für ein  $i \in \left\{ -\left\lfloor \frac{N-1}{m} \right\rfloor, \dots, \left\lfloor \frac{N-1}{m} \right\rfloor \right\}$

**Behauptung:** Für jedes  $i$  ex. höchstens ein  $k$  mit  $g(k) = im$ .

Es folgt  $|\{k : h_k(x) = h_k(y)\}| \leq 2 \lfloor (N-1)/m \rfloor$  und somit

$$\sum_{k=1}^{N-1} \sum_{i=0}^{m-1} b_{ik} (b_{ik} - 1) \leq 2 \lfloor \frac{N-1}{m} \rfloor n(n-1)$$

# Beweis der Behauptung

---

Seien  $k_1, k_2 \in \{1, \dots, N-1\}$  mit  $g(k_1) = g(k_2)$

$$g(k_1) = g(k_2)$$

$$\Leftrightarrow (k_1 x) \bmod N - (k_1 y) \bmod N = (k_2 x) \bmod N - (k_2 y) \bmod N$$

$$\Rightarrow ((k_1 x - k_1 y) - (k_2 x - k_2 y)) \bmod N = 0$$

$$\Leftrightarrow (k_1 - k_2)(x - y) \bmod N = 0$$

Also  $(k_1 - k_2) \bmod N = 0$  oder  $(x - y) \bmod N = 0$ .

Somit  $k_1 = k_2$ .

# Folgerungen

**Korollar 1:** Es gibt mindestens  $(N-1)/2$  viele  $k$  mit  $B_k \leq 4n(n-1)/m$ .  
Ein solches  $k$  an in erwarteter Zeit  $O(m+n)$  bestimmt werden.

**Beweis:** Sei  $A = \{ k : B_k > 4n(n-1)/m \}$

Annahme:  $|A| > (N-1)/2$

$$\Rightarrow \sum_{k=1}^{N-1} B_k \geq \sum_{k \in A} B_k > \frac{N-1}{2} \frac{4n(n-1)}{m} = \frac{N-1}{m} 2n(n-1)$$

Mit  $WSK \geq \frac{1}{2}$  erfüllt ein zufällig gewähltes  $k$  die Bedingung. Die erwartete Anzahl der Versuche ist  $\leq 2$ .

## Korollar 2:

- a) Sei  $m = 2n(n-1)+1$ . Dann sind mindestens  $(N-1)/2$  der  $h_k$  injektiv auf  $S$ . Ein solches  $h_k$  findet man in erwarteter Zeit  $O(m+n)=O(n^2)$ .
- b) Sei  $m = n$ . Dann gilt für mindestens  $(N-1)/2$  der  $h_k$ , dass  $B_k \leq 4(n-1)$ . Ein solches  $h_k$  findet man in erwarteter Zeit  $O(n)$ .

**Beweis:** Wegen Korollar 1 gibt es  $(N-1)/2$  viele  $h_k$  mit

$$B_k \leq \frac{4n(n-1)}{2n(n-1)+1} < 2.$$

Wegen Lemma 1 ist  $h_k$  injektiv.

# Zweistufiges Schema

$$S \subseteq U = [0 \dots N-1] \quad |S| = n = m$$

**Idee:** Wende Kor. 2b an und teile  $S$  in Teilmengen der Größe  $O(\sqrt{n})$ .  
Auf jede Teilmenge wende Kor. 2a an.

1. Wähle  $k$  mit  $B_k \leq 4(n-1) \leq 4n$ .

$$h_k : x \rightarrow ((kx) \bmod N) \bmod n$$

2.  $W_i = \{x \in S : h_k(x) = i\}$ ,  $b_i = |W_i|$ ,  $m_i = 2b_i(b_i-1)+1$  für  $1 \leq i \leq n-1$

Wähle  $k_i$  so, dass

$$h_{k_i} : x \rightarrow (k_i x \bmod N) \bmod m_i$$

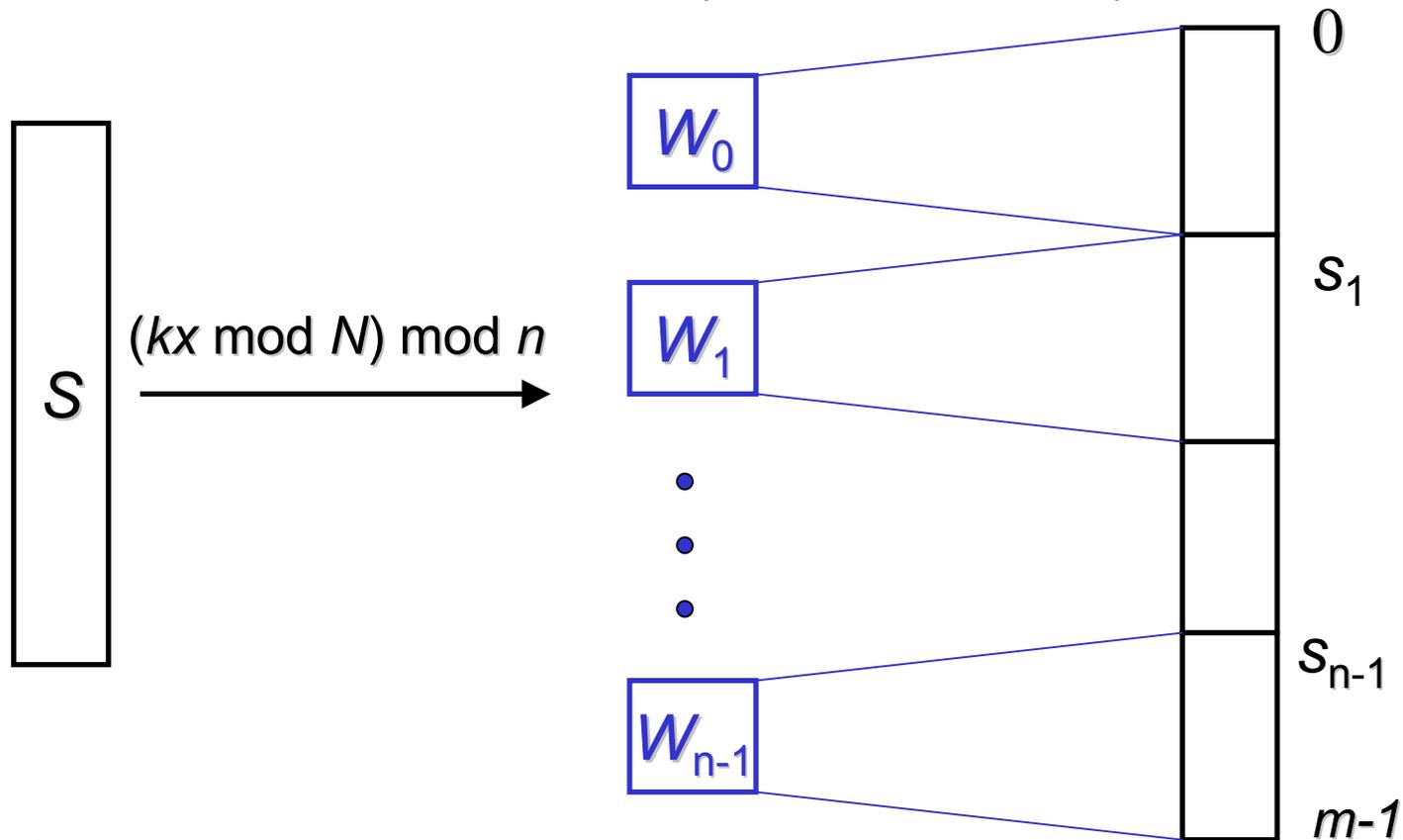
eingeschränkt auf  $W_i$  injektiv ist.

# Zweistufiges Schema

$$3. s_i = \sum_{j < i} m_j$$

Speichere  $x \in S$  in Tafelposition  $T[s_j + j]$  wobei

$$i = (kx \bmod N) \bmod n \quad j = (k_i x \bmod N) \bmod m_i$$



$$m = \sum_{i=0}^{n-1} m_i = \sum_{i=0}^{n-1} (2b_i(b_i - 1) + 1) = n + 2B_k$$
$$\leq n + 8(n - 1) \leq 9n$$

Zusätzlich braucht man Platz für die  $k_i$ ,  $m_i$  und  $s_i$ .  
Platzbedarf insgesamt  $O(n)$ .

# Zeitbedarf für den Aufbau

---

- Nach Kor. 2b kann  $k$  in erwarteter Zeit  $O(n)$  gefunden werden.
- Die  $W_i, b_i, m_i, s_i$  können in Zeit  $O(n)$  berechnet werden.
- Nach Kor. 2a kann jedes  $k_i$  in erwarteter Zeit  $O(b_i^2)$  berechnet werden.

Erwartete Gesamtlaufzeit:

$$O\left(n + \sum_{i=0}^n b_i^2\right) = O(n + B_k) = O(n)$$

# Hauptergebnis

---

**Satz:** Sei  $N$  eine Primzahl und  $S \subseteq U = [0 \dots N-1]$  mit  $|S| = n$ .  
Für  $S$  kann eine **perfekte Hash-Tafel** der Größe  $O(n)$  und eine Hash-Funktion mit Zugriffszeit  $O(1)$  in **erwarteter Zeit  $O(n)$**  aufgebaut werden.