Algorithmen und Datenstrukturen (ESE)
Entwurf, Analyse und Umsetzung von Algorithmen (IEMS)
WS 2012 / 2013

Vorlesung 5, Dienstag 20. November 2012
(Wie baut man eine Hash Map, Universelles Hashing)

Prof. Dr. Hannah Bast
Lehrstuhl für Algorithmen und Datenstrukturen
Institut für Informatik
Universität Freiburg

UNI FREIBURG

Blick über die Vorlesung heute

UNI FREIBURG

- Organisatorisches
  - Ihre Erfahrungen mit dem Ü4 (Map / GeoNames)
  - Treffen mit Ihrem Tutor / Ihrer Tutorin
  - Tipp für Windows Benutzer: cygwin
- Hash Maps
  - Eine mögliche Realisierung von einer Map
  - Dabei zentral: **universelles Hashing**
  - Beispiele für universelle Klassen von Hashfunktionen
  - Übungsblatt: Mittels eines Programms nachprüfen, dass eine Klasse von Hashfunktionen universell ist
  - **Achtung:** dazu kam in den letzten Prüfungen jedes Mal eine Aufgabe, und wenige haben es ganz richtig gemacht !

Erfahrungen mit dem Ü4 (Map / GeoNames)

UNI FREIBURG

- Zusammenfassung / Auszüge Stand 20. November 16:13
  - War für viele sehr / zu zeitintensiv, Gründe:

    Die HashMap<string, Hashmap<...>> hat einige verwirrt

    Ebenso das Sortieren mit Collections.sort ... trotz Vormachen !

    Vor allem: fehlende Programmiererfahrung
  - Manche haben daher die Variante "mit Sortieren" weggelassen
  - Vorschlag zur Güte: Maximalpunktzahl für Ü4 → 10 Punkte

    die, die alles gemacht haben, haben dann 10 Bonuspunkte
  - Den Vorzug einer Map schätzen gelernt
  - Live-Coding in der Vorlesung schneller als man schauen kann
  - Weniger vorgeschrieben bekommen ... die meisten wollen das !

Treffen mit Ihrem Tutor / Ihrer Tutorin

UNI FREIBURG

- Grund
  - Wir wollen Sie alle mal kennen lernen
  - Die meisten fanden das in der Vergangenheit gut
  - Gelegenheit für Fragen, die man sonst nicht stellt
  - Wir wollen auch schauen, dass es Sie wirklich gibt und Sie die Übungsblätter im Wesentlichen selber machen
- Vorgehen
  - Sie werden von Ihrem Tutor / Ihrer Tutorin angeschrieben
  - Treffen dauert ca. 30 Minuten
  - Ein Treffen pro Semester ist Pflicht, für alle !

Tipp für Windows-Benutzer

UNI FREIBURG

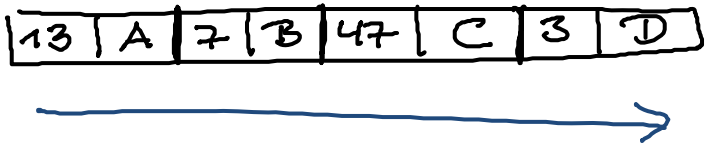
- Cygwin
  - Download unter www.cygwin.com
  - Dann haben Sie in Ihrer normalen DOS shell auch alle bekannten Unix/Linux Befehle
  - Insbesondere: cut, head, tail, less, more, sort, uniq, ...

Wie baut man eine Map?

UNI FREIBURG

- Zur Erinnerung
  - Ein assoziatives Array ist wie ein normales Array, nur dass die Indizes nicht 0, 1, 2, ... sind, sondern irgendwas
- Problem
  - Schnell ein Element mit einem bestimmten Schlüssel finden
  - Naive Lösung: Paare von Schlüsseln und Werten in einem normalen Feld (Java: ArrayList, C++: vector) speichern

    Array<KeyValuePair>
  - Bei n Schlüsseln kostet die Suche dann bis zu Θ(n) Zeit
  - Mit einer Hash Map geht es im günstigsten Fall in Zeit Θ(1) ... und zwar egal wieviele Elemente schon in der Map sind!



HashMap — Grundidee

UNI FREIBURG

- Grundidee
  - Abbildung der Schlüssel auf die Indizes von einem normalen Feld, mit Hilfe einer sogenannten Hashfunktion
- Ein einfaches Beispiel
  - Schlüsselmenge { 312692, 3904433, 5148949 }
  - Hashfunktion $h(x) = x \bmod 5$, also Wertebereich [0..4]
    - $h(312692) = 2$, $h(3904433) = 3$, $h(5148949) = 4$
  - Ein gewöhnliches Feld T der Größe 5 (die Hashtabelle)
  - Wir speichern das Element mit Schlüssel x in $T[h(x)]$
  - In unseren Beispiel jetzt Zugriff in $\Theta(1)$ Zeit
  - Problem: zwei Schlüssel mit $x \neq y$ aber $h(x) = h(y)$
  - Das nennt man **Kollision**

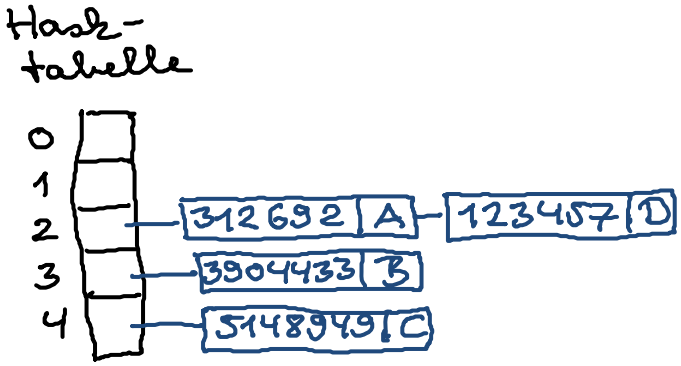
HashMap — Kollisionen

- Einfache Lösung

  $h(x) = x \bmod 5$

  – Jeder Eintrag der Hashtabelle kann nicht nur ein key-value Paar speichern, sondern eine Menge davon

  Array<Array<KeyValuePair>> hashTable;



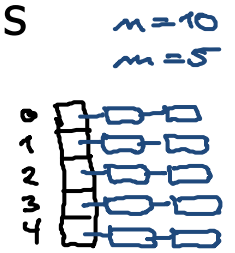
Schlüssel 312692 und 123457 „kollidieren"

insert (312692, A)
$h(312692) = 2$
insert (3904433, B)
$h(3904433) = 3$
insert (5148949, C)
$h(5148949) = 4$
insert (123457, D)
$h(123457) = 2$
lookup (123457) ?
$h(123457) = 2 \rightarrow$ D
lookup (123459) ?
$h(123459) = 4 \rightarrow$ GIBT'S NICHT

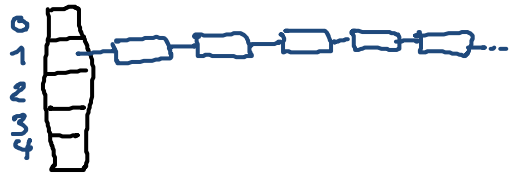
usw...

HashMap — Kollisionen

- Laufzeit für die Schlüsselsuche
  - Im besten Fall werden die Schlüssel gleichmäßig auf das Feld verteilt
    - Bei n Schlüsseln und einer Hashtabelle der Größe m sind das dann $\approx n/m$ Schlüssel pro Eintrag



  - Im schlechtesten Fall werden alle Schlüssel auf denselben Eintrag der Hashtabelle abgebildet
    - Dann sind wir wieder bei Zeit $\Theta(n)$



- Lösung
  - Wir wählen die Hashfunktion zufällig aus einer geeigneten Menge von Hashfunktionen, so dass die Schlüssel im Erwartungsfall gleichmäßig verteilt sind
  - Das nennt man **universelles Hashing**

Universelles Hashing 1/7

UNI FREIBURG

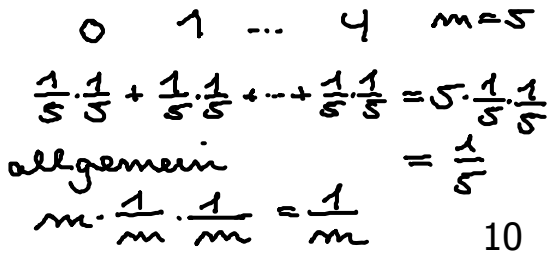
- Definition
  - Sei U die Menge der möglichen Schlüssel (Universum) und sei m die Größe der Hashtabelle
  - Sei H eine Menge von Hashfunktionen $U \rightarrow \{0, \ldots, m-1\}$
  - H ist c-universell wenn für alle $x, y \in U$ mit $x \neq y$ gilt:

    $| \{h \in H : h(x) = h(y)\} | \leq c \cdot |H| / m$
  - Mit anderen Worten, wenn $h \in H$ zufällig gewählt, dann

    $Prob(h(x) = h(y)) \leq c \cdot 1 / m$

    also: c=1 besser geht's nicht.
  - Bemerkung: wenn man x und y jede **zufällig** in eine der m "slots" der Hashtabelle schmeißt, dann

    $Prob(Kollision) = 1 / m$



Universelles Hashing 2/7

UNI FREIBURG

- Satz
  - Sei H eine c-universelle Klasse von Hashfunktionen
  - Sei S eine Menge von Schlüsseln und $h \in H$ zufällig gewählt
  - Sei S_i die Menge der Schlüssel x mit $h(x) = i$
  - Dann ist $E(|S_i|) \leq 1 + c \cdot |S| / m$ für alle i
  - Insbesondere: Falls $m = \Omega(|S|)$ gilt $E(|S_i|) = O(1)$

ideal wäre $|S_i| = |S|/m$ für alle i

Bevor wir das beweisen, ein kleiner Auffrisch- bzw. Crash-Kurs in Wahrscheinlichkeitsrechnung

Einschub: Wahrscheinlichkeitsrechnung 1/3

UNI FREIBURG

- Wahrscheinlichkeitsraum / Ereignisse
  - Wir beschränken uns hier auf den diskreten Fall
  - Wahrscheinlichkeitsraum Ω von sog. Elementarereignissen
  - Die haben Wahrscheinlichkeiten ... Bedingung $\Sigma_{e \in \Omega} \Pr(e) = 1$
  - Ereignis E = Teilmenge von Ω, Wahrsch. $\Pr(E) = \Sigma_{e \in E} \Pr(e)$
  - Zum Beispiel: zweimal würfeln, dann $\Omega = \{1,...,6\}^2$
    - Jedes e aus Ω hat dann Wahrscheinlichkeit $\Pr(e) = 1/36$
    - E = beide Augenzahlen sind gerade, dann $\Pr(E) = 9/36 = 1/4$

(2,2) (2,4) (2,6) (4,2) (4,4) (4,6) (6,2) (6,4) (6,6)

Einschub: Wahrscheinlichkeitsrechnung 2/3

UNI FREIBURG

- Zufallsvariable
  - ... weist einem Ausgang des Zufallsexperiments eine Zahl zu
  - Zum Beispiel: X = Summe Augenzahlen bei zweimal Würfeln
  - Sowas wie X = 12 oder X ≥ 7 sind dann einfach Ereignisse
  - Beispiel 1: Prob(X=2) = $\frac{1}{36}$ (1,1)
  - Beispiel 2: Prob(X=4) = $\frac{3}{36} = \frac{1}{12}$ (1,3) (2,2) (3,1) (1,4) (2,3) (3,2) (4,1)
  - **Erwartungswert** ist definiert als **E**(X) = Σ k · Pr(X = k)

    Intuitiv: gewichtetes Mittel der möglichen Werte von X, wobei die Gewichte die Wahrscheinlichkeiten der entspr. Werte sind

einmal würfeln: $1+\dots+6 = 6\cdot 7/2 = 21$

$$E(X) = \frac{1}{6}\cdot 1 + \frac{1}{6}\cdot 2 + \frac{1}{6}\cdot 3 + \frac{1}{6}\cdot 4 + \frac{1}{6}\cdot 5 + \frac{1}{6}\cdot 6 = 21/6 = 3.5$$

zweimal würfeln:

$$E(X) = \frac{1}{36}\cdot 2 + \frac{2}{36}\cdot 3 + \frac{3}{36}\cdot 4 + \frac{4}{36}\cdot 5 + \dots = 7$$

Einschub: Wahrscheinlichkeitsrechnung 3/3

$EX_1 = 3.5$
$EX_2 = 3.5$

X_1 = AZ Würfel 1
X_2 = AZ Würfel 2
$X = X_1 + X_2$ $E(X) = ?$

- Summe von Erwartungswerten
  - Für beliebige (diskrete) Zufallsvariablen $X_1, \ldots, X_n$ gilt

    $\mathbf{E}(X_1 + \ldots + X_n) = \mathbf{E}(X_1) + \ldots + \mathbf{E}(X_n)$

    $E(X) = 3.5 + 3.5 = 7$
  - Korollar: Bei einem Zufallsexperiment tritt das Ereignis E mit Wahrscheinlichkeit p auf. Sei X die Anzahl der Auftreten von E bei n Ausführungen dieses Experimentes, dann ist $\mathbf{E}(X) = n \cdot p$
  - Beispiel: $\mathbf{E}$(Anzahl Sechser bei 60 mal Würfeln) = 10
  - Beweis Korollar:

$X_i = \begin{cases} 1, & \text{falls Ereignis eintritt} \\ 0, & \text{sonst} \end{cases}$

das nennt man INDIKATORVARIABLE

$\Rightarrow X = \sum_{i=1}^{n} X_i$ = # der Male, wo das Ereignis eintritt

$E(X) = E\left(\sum_{i=1}^{n} X_i\right) \overset{\text{Satz oben}}{=} \sum_{i=1}^{n} E(X_i) = \sum_{i=1}^{n} p = n \cdot p$ ∎

//Def. E-Wert

$0 \cdot Pr(X_i = 0) + 1 \cdot Pr(X_i = 1) = Pr(X_i = 1) = p$

Universelles Hashing 3/7

- Beweis von $E(|S_i|) \leq 1 + c \cdot |S| / m$ für alle i

$S_i = \{x \in S \mid h(x) = i\}$

Sei $x \in S_i$ irgendein Schlüssel → falls $S_i = \emptyset \Rightarrow |S_i| = 0$

Definiere $I_y = \begin{cases} 1, & \text{falls } h(y) = i \\ 0, & \text{sonst} \end{cases}$ für alle $y \in S \setminus \{x\}$

$$\Rightarrow |S_i| = 1 + \sum_{y \in S \setminus \{x\}} I_y$$

$$\Rightarrow E(|S_i|) = E\left(1 + \sum_{y \in S \setminus \{x\}} I_y\right) = 1 + \sum_{y \in S \setminus \{x\}} E(I_y) = (*)$$

$$(*) \leq 1 + \sum_{y \in S \setminus \{x\}} c \cdot \frac{1}{m}$$

$$= 1 + (|S| - 1) \cdot c \cdot \frac{1}{m} \quad (|S| - 1 \leq |S|)$$

$$\leq 1 + c \cdot |S| / m \quad \blacksquare$$

$E(I_y) = Pr(I_y = 1)$
$= Pr(h(y) = i)$
$= Pr(h(y) = h(x))$
$\leq c \cdot \frac{1}{m}$
nach Definition von Universalität

Universelles Hashing 4/7

- Negativbeispiel 1
  - Die Menge aller h mit $h(x) = a \cdot x \bmod m$, für ein $a \in U$
  - Ist nicht universell, warum?

falls universell: $\forall x,y\ x \neq y \quad |\{h : h(x) = h(y)\}| \leq c \cdot \frac{|H|}{m}$

ABER: $x = m$ und $y = 2m$

Dann $x \neq y$, und $h(x) = a \cdot x \bmod m = 0$ ✓a

$h(y) = a \cdot y \bmod m = 0$ ✓a

$\Rightarrow |\{h : h(x) = h(y)\}| = |H|$

- Negativbeispiel 2
  - Die Menge aller Funktionen von $U \to \{1,\ldots,m\}$
  - Ist 1-universell, warum?

Weil $Pr(x \text{ und } y \text{ kollidieren bei zufälligen Werten}) = \frac{1}{m}$

$= c \cdot \frac{1}{m}$ mit $c = 1$

  - Aber als Hashfunktionen ungeeignet, warum?

man muss sich für jeden Schlüssel einzeln merken, wo man ihn „hingeworfen" hat

$\Rightarrow$ Platz und Zeit $O(|S|)$

Universelles Hashing 5/7

$U = \{0, \ldots, 999\}$
$p = 101$ (prim)

$a = 47$
$b = 5$
$h_2(x) = (47x + 5) \bmod 101 \bmod m$

UNI FREIBURG

- Positivbeispiel 1
  - Sei p eine große Primzahl, und zwar $p > m$ und $p \geq |U|$
  - Sei H die Menge aller h mit $h(x) = (a \cdot x + b) \bmod p \bmod m$

    wobei $a, b \in U$
  - Die ist **≈1**-universell, siehe Exercise 4.11 in Mehlhorn/Sanders

Universelles Hashing 6/7

$U = \{0, \ldots, 999\}$

$m = 10$

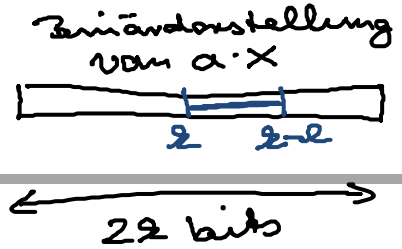
$x = 127$

$a = 348$

$a \bullet x = 1 \cdot 3 + 2 \cdot 4 + 7 \cdot 8$

- Positivbeispiel 2
  - Die Menge aller h mit $h(x) = a \bullet x \bmod m$, für ein $a \in U$
    - Schreibe $a = \Sigma_{i=0..k-1} a_i \cdot m^i$, wobei $k = \text{ceil}(\log_m |U|)$
    - Entsprechend $x = \Sigma_{i=0..k-1} x_i \cdot m^i$
    - Dann $a \bullet x := \Sigma_{i=0..k-1} a_i \cdot x_i$
    - Intuitiv: das "Skalarprodukt" der Darstellung zur Basis m
  - Die ist **1**-universell, siehe Theorem 4.4 in Mehlhorn/Sanders

Universelles Hashing 7/7



UNI FREIBURG

- Positivbeispiel 3
  - Die Menge aller h mit $h(x) = a \cdot x \bmod 2^k \operatorname{div} 2^{k-\ell}$ für $a \in U$
    - ... wobei $|U| = 2^k$, $m = 2^\ell$ ... in der Regel $k >> \ell$
    - Das $\cdot$ ist hier wieder das normale Produkt
    - Das heißt $a \cdot x$ gibt eine Zahl aus $0..|U|^2$
    - Die lässt sich also in Binärdarstellung mit $2k$ Bits darstellen
    - Eine Position in der Hashtabelle lässt sich mit ℓ Bits darstellen
    - $h(x)$ ist dann einfach der Wert der Bits $k-l..k-1$ von $a \cdot x$
  - Diese Menge von Hashfunktionen ist **2**-universell
  - Siehe Exercise 4.14 in Mehlhorn / Sanders

Literatur / Links

UNI FREIBURG

- Hash Maps
  - In Mehlhorn/Sanders:

    4 Hash Tables and Associative Arrays
  - In Cormen/Leiserson/Rivest

    12 Hash Tables
  - In Wikipedia

    http://de.wikipedia.org/wiki/Hashtabelle

    http://en.wikipedia.org/wiki/Hash_table
- Hash Map in Java und in C++

  http://download.oracle.com/javase/1.4.2/docs/api/java/util/HashMap.html

  http://www.sgi.com/tech/stl/hash_map.html