Informatik II: Algorithmen und Datenstrukturen SS 2017

Vorlesung 5b, Mittwoch, 24. Mai 2017
(Universelles Hashing Teil 2, Perfektes Hashing)

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

Blick über die Vorlesung heute

- Drumherum
  - Mathe-Phobie — Ursachen und Heilung
- Inhalt
  - Universelle Klassen von Hash-Funktionen — Zwei Negativ- und drei Positivbeispiele
  - Worst-Case Komplexität — Gestern: average-case
  - Perfektes Hashing — Wenn S vorher bekannt
  - Histogramme — Für das ÜB5

Mathe-Phobie 0.25

- Kommentare aus den erfahrungen.txt
  - Wie bei Spinnen: zu viele Beine aka Unbekannte
  - 2 Punkte im Mathe-Abi ist doch besser als nichts
  - Ich habe schon als kleines Kind nicht gerne geteilt
  - Gründe 1: Mathe, Mathe 1, Mathe 2, schlechte Lehrer, …
  - Gründe 2: Vorurteile, mangelnde Motivation, ungenügende geistige Ausdauer (in dieser Reihenfolge)
  - Nicht post-faktisch genug: man kann sich nicht rauslabern … wenn es falsch ist, ist es falsch, kaum einer hat gerne Unrecht
  - Angst vor dem Scheitern; es gibt nur richtig oder falsch, "ein bisschen richtig" wie bei anderen Fächern gibt es nicht

Mathe-Phobie 0.50

- Aus der Neuropsychologie
  - Es gibt zwei Arten von Zahlensystemen im Gehirn:
  - **Object tracking system (OTS):** funktioniert über die "parallele" Wahrnehmung von verschiedenen Objekten

    Grundlage unserer internen Darstellung der Zahlen 0, 1, 2, 3 (bei Kindern) bzw. 0, 1, 2, 3, 4 (bei Erwachsenen)
  - **Approximate number system (ANS):** Abschätzung und Vergleich der Größenordnungen von Gruppen

    Erwachsene können damit "auf einen Blick" Unterschiede in Gruppengrößen von ca. 10 – 15% erkennen

    Kleine Kinder erst Unterschiede von Faktor 2 oder größer … wird dann bis zum Erwachsenenalter immer trennschärfer

Mathe-Phobie 0.75

- Aus der Neuropsychologie, Fortsetzung
  - OTS und ANS sind evolutionär beides sehr alte Systeme

    Gibt es auch schon bei Tieren, sogar schon bei **Insekten**
  - Beim Menschen viel Forschung zum Zusammenhang zwischen Präzision des ANS und späteren mathematischen Fähigkeiten

    Libertus et al: "Preschool acuity of the approximate number system correlates with school math ability", Development 2011

    Ergebnis der Studie: die Genauigkeit des ANS im Vorschulalter ist stark mit Matheleistungen im Grundschulalter korreliert

    Das sollte aber keine Entschuldigung sein, siehe nächste Folie

Mathe-Phobie 1.0

- Von der Mathe-Phobie zur Mathe-Philie
  - Selbst bei ungünstigen ANS-Voraussetzungen, kann jeder etwas für eine gesunde Mathe-Philie tun

    Es gibt auch Studien, die zeigen, dass im weiteren Verlauf der Entwicklung nicht die Grundintelligenz sondern die **Übung** der entscheidende Faktor ist

  - Die "Phobie" kann tatsächlich real sein und aktiviert in der Amygdala dieselben Reflexe wie bei Schmerz und Angst

    Das Gehirn ist dann mit der Verarbeitung der Stress-Situation beschäftigt anstatt mit dem mathematischen Problem

    Wichtig für den Umgang damit: Umfeld zum Üben schaffen ohne kritischen Gegenüber und vor allem **ohne Zeitstress**

Klassen von Hashfunktionen 1/8

- Vorbetrachtung
  - Im folgenden sind vorgegeben:

    die Größe m der Hashtabelle

    das Universum U und seine Größe |U|

    Das weiß man ja in der Praxis beides, bevor man eine Hash-Funktion auswählt

Klassen von Hashfunktionen 2/8

x und y kollidieren nur für einen Bruchteil der Hash-Funktionen

- Negativbeispiel 1
  - Alle h mit $h(x) = a \cdot x \bmod m$, mit $a \in \{0, \ldots, m-1\}$

    Das sind m verschiedene Hashfunktionen
  - Diese Klasse von Funktionen ist **nicht** universell
  - Zum Beweis erst noch mal die Definition von universell:

    Für alle $x \neq y$ ist $|\{ h \in H : h(x) = h(y) \}| \leq c \cdot |H| / m$
  - Für die Nicht-Universalität reicht also ein Schlüsselpaar x, y mit $x \neq y$ für dass das nicht gilt:

zum Beispiel: $x = m$ und $y = 2m$

dann $\forall a \in \{0, \ldots, m-1\}$: $h(x) = a \cdot x \bmod m$

$= a \cdot m \bmod m = 0$

$h(y) = a \cdot 2m \bmod m = 0$

$\Rightarrow |\{h \in H : h(x) = h(y)\}| = |H|$ $\Rightarrow$ nicht universell!

alle $h \in H$ sind schlecht für diese x und y

Klassen von Hashfunktionen 3/8

- Negativbeispiel 2
  - Die Menge H **aller** Funktionen von U → {0, ..., m – 1}

    die Menge aller möglichen Schlüssel

    Das sind ganz schön viele, nämlich: $\underbrace{m \cdot m \cdot \ldots \cdot m}_{|U| \text{ mal}} = m^{|U|}$

  - Für eine zufällige Funktion h ϵ H ist dann für jedes x ϵ U h(x) zufällig aus {0, ..., m – 1}

    Eine zufällige Funktion h aus H wählen und eine Menge S damit abbilden ist also wie zufälliges Werfen

    siehe VL 5a von gestern

  - Für zufälliges Werfen gilt: Pr(h(x) = h(y)) = 1/m
  - Die Klasse ist also 1-universell

    Besser geht es nicht, warum dann Negativbeispiel ?

Klassen von Hashfunktionen 4/8

- Negativbeispiel 2, Fortsetzung
  - Als Klasse von Hashfunktionen trotzdem ungeeignet
  - Die Funktionen haben keine "kompakte" Form

    So wie zum Beispiel $a \cdot x \bmod m$
  - Um eine Funktion h aus H zu repräsentieren, müsste man für jedes $x \in U$ speichern, was $h(x)$ ist

    Das braucht $\Theta(|U|)$ Platz
  - Es würde auch reichen, den Wert von $h(x)$ nur für $x \in S$ zu speichern, aber dafür bräuchte man … eine Map

    Das kommt öfter vor: man trifft bei einem Lösungs-versuch wieder auf das ursprüngliche Problem

Klassen von Hashfunktionen 5/8

→ falls $p \gg m$, dann c sehr nahe an 1. falls m prim sogar $c = 1$

genauer: $c = \dfrac{\lceil p/m \rceil}{p/m}$

man braucht nicht $p \geq |U|$ wie gestern behauptet

- Positivbeispiel 1
  - Sei p eine Primzahl mit p > m und U = {0, ..., p – 1}

    p ist fix für die Klasse
  - Sei H die Menge aller Hash-Funktionen h mit

    entscheidend eigentlich: $\mathrm{ggT}(m, p) = 1$

    h(x) = (a · x + b) mod p mod m ... wobei a, b ϵ {0, ..., p – 1}
  - Diese Menge von Hashfunktionen ist **≈1**-universell

    Beweis siehe Exercise 59 in Mehlhorn/Sanders ... nicht K
  - Intuition: man kann sich klarmachen, wie das mod p vor dem mod m die Probleme vom Negativbeispiel 1 verhindert

    Beispiel: m = 10, p = 101, x = 80, y = 30, a = 2, b = 17

mit Negativ-Klasse 1: $h(x) = \underbrace{a}_{=2} \cdot \underbrace{x}_{=80} \,\%\, \underbrace{m}_{=10} = 0$; $h(y) = \underbrace{a}_{=2} \cdot \underbrace{y}_{=30} \,\%\, \underbrace{m}_{=10} = 0$

mit der Klasse oben: $h(x) = (\overbrace{\underbrace{a}_{=2} \cdot \underbrace{x}_{=80} + \underbrace{b}_{=17}}^{177}) \bmod \underbrace{p}_{=101} \bmod \underbrace{m}_{=10} = 76 \,\%\, 10 = 6$

$h(y) = (\underbrace{a}_{=2} \cdot \underbrace{y}_{=30} + \underbrace{b}_{=17}) \bmod \underbrace{p}_{=101} \bmod \underbrace{m}_{=10} = 77 \,\%\, 10 = 7$

Klassen von Hashfunktionen 6/8

- Positivbeispiel 2
  - Die Menge aller h mit $h(x) = a \bullet x \bmod m$

    wobei $a \in U$ und m eine **Primzahl** sein muss

    Zusatzaufgabe: finde die nächste Primzahl $\geq m$
  - Die Operation $\bullet$ ist dabei wie folgt definiert:

    Schreibe $a = \Sigma_{i=0..k-1}\, a_i \cdot m^i$, wobei $k = \lceil \log_m |U| \rceil$

    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: $a \bullet x$ ist quasi das "Skalarprodukt" der Darstellung von a und x zur Basis m ... Rechenbeispiel siehe nächste Folie

Klassen von Hashfunktionen 7/8

- Positivbeispiel 2, Fortsetzung
  - Diese Klasse ist sogar **1**-universell

    Beweis siehe Theorem 12 in Mehlhorn/Sanders …

  - Beispiel für eine Hashfunktion aus dieser Klasse mit:

    m = 11, U = {0, …, 11^3 – 1}, a = 274, x = 47

zur Basis 10:

$a = 274 = 2 \cdot \underbrace{10^2}_{=10^2} + 7 \cdot 10 + 4 \cdot \underbrace{1}_{=10^0}$

$a = 274 = \underbrace{2 \cdot 11^2}_{=242} + \underbrace{2 \cdot 11}_{=22} + \underbrace{10 \cdot 1}_{=10} = (2\ 2\ 10)$ zur Basis 11

$x = 47 = \underbrace{0 \cdot 11^2}_{=0} + \underbrace{4 \cdot 11}_{=44} + 3 \cdot 1 = (0\ 4\ 3)$ zur Basis 11

$a \bullet x = (2\ 2\ 10) \bullet (0\ 4\ 3) = 2 \cdot 0 + 2 \cdot 4 + 10 \cdot 3 = 38$

$a \bullet x \bmod m = 38 \bmod 11 = 5$

Klassen von Hashfunktionen 8/8

gibt ℓ Bits = eine Zahl aus $0..2^\ell-1$ ($m-1$)

a : $k-1 \ldots 0$ — k Bits

x : $k-1 \ldots 0$ — k Bits

$a \cdot x$: $2k-1 \ldots k-1 \;\; k-\ell \ldots 0$ — $2k$ Bits

- Positivbeispiel 3
  - Die Menge H aller h mit $h(x) = a \cdot x \bmod 2^k \operatorname{div} 2^{k-\ell}$

    wobei $U = \{0, \ldots, 2^k - 1\}$, $a \in U$ **ungerade** und $m = 2^\ell$

    $\in 0..2^k-1$ $\quad$ $\in 0..2^k-1$ $\quad$ sogar: $(2^k-1)^2 < 2^{2k}-1$
  - Das (normale) Produkt $a \cdot x$ gibt eine Zahl aus $0 .. 2^{2k} - 1$
  - In Binärdarstellung: $a \cdot x$ lässt sich mit $2k$ Bits darstellen, eine Position in der Hashtabelle mit ℓ Bits

    zum Beispiel: $15 \gg 2 = 3 = \lfloor \frac{15}{2^2} \rfloor$ (1111, 11)
  - $h(x)$ ist dann einfach der Wert der Bits $k-\ell..k-1$ von $a \cdot x$

    << und >>

    Für das ÜB5: mit Bitschiebe-Operationen ausschneiden
  - Diese Menge von Hashfunktionen ist **2**-universell

    Siehe Exercise 62 in Mehlhorn / Sanders

Worst-Case Komplexität 1/3

- Bisher: Average-Case Komplexität
  - Wenn h aus einer universellen Klasse gewählt wird, dann ist für jeden Schlüssel x aus S: $E(|S_x|) \leq 1 + c \cdot |S| / m$

    Insbesondere, wenn $m = \Theta(|S|)$: $E(|S_x|) = O(1)$
  - Sei S_{max} der Platz in der Hashtabelle, auf den die meisten Schlüssel abgebildet werden, gilt dann auch

    $E(|S_{max}|) = O(1)$?
  - Im Allgemeinen: $E(\max\{X_1, \ldots, X_n\}) \neq \max\{E(X_1), \ldots, E(X_n)\}$

    E und Summe kann man vertauschen, E und max nicht

Worst-Case Komplexität 2/3

- Zufälliges Werfen
  - Nehmen wir an, wir werfen zufällig n Bälle in m Kisten

    Besser kann eine zufällige Hashfunktion nicht verteilen
  - Sei S_i die Menge der Bälle in der i-ten Kiste und sei S_{max} die Kiste mit den meisten Bällen
  - Dann ist $E(|S_i|) = n / m$
  - Aber auch in dem Fall **nicht** $E(|S_{max}|) = n / m$

    Dazu schreiben wir gerade ein kleines Programm …

Worst-Case Komplexität 3/3

- Satz
  - Nehmen wir an, wir werfen zufällig n Bälle in n Kisten und seien S_i und S_{max} wie auf der Folie vorher

    Dann ist $|S_{max}| = O(\log n / \log \log n)$ mit hoher W-keit
  - Wenn man n Bälle in m Kisten wirft, mit $n \geq m \cdot \log m$

    Dann ist $|S_{max}| = \Theta(n / m)$ mit hoher W-keit
  - Der Beweis würde hier zu weit führen … nur kurz, wie der Term $\log n / \log \log n$ überhaupt zustande kommt:

    $k^k = n \quad \Rightarrow \quad k \approx \log n / \log \log n$ Beweis: zu Hause

    zum Vergleich: $2^x = n \Rightarrow x = \log_2 n$

Perfektes Hashing 1/4

- Vorbetrachtung
  - Gegeben eine Schlüsselmenge S
  - Die Größe der Hashtabelle m sei $\geq |S|$
  - Dann gibt es (sogar viele) Hashfunktionen h, die S ohne Kollisionen (also perfekt) auf $\{0, \ldots, m-1\}$ abbilden

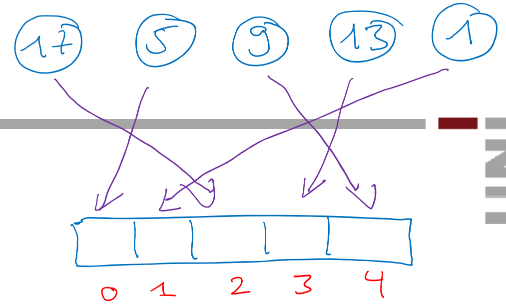
    Mathematisch gleichbedeutend mit: h ist **injektiv**
  - Allerdings kann man diese Funktionen nicht unbedingt in wenig Platz speichern … siehe Negativbeispiel 2

    Man kann aber Funktionen finden, für die das geht

Perfektes Hashing 2/4

- Definition
  - Gegeben eine Schlüsselmenge S
  - Die Größe der Hashtabelle m sei $\geq |S|$
  - Eine Hashfunktion h heißt **perfekt** für S, wenn gilt:
    - h bildet S injektiv auf $\{0, \ldots, m-1\}$ ab
    - h kann in $O(m)$ Platz gespeichert werden
    - $h(x)$ kann für alle x in $O(1)$ ausgewertet werden

Perfektes Hashing 3/4



- Beispiel
  - Sei $S = \{17, 5, 9, 13, 1\}$ und $m = 5$
  - Dann wäre folgende Hashfunktion perfekt:

    $h(x) = x \bmod 5$
  - Kann man immer so eine einfache Funktion finden?
  - Antwort: so einfach nicht immer, aber einfach genug!

    Achtung: das klappt wohlgemerkt nur, wenn die gesamte Menge S bekannt ist, bevor wir eine Map bauen

    Das ist manchmal, aber oft auch nicht der Fall

Perfektes Hashing 4/4

- Satz
  - Sei S eine beliebige Schlüsselmenge und $m \geq 2 \cdot |S|$

    Dann gibt es eine perfekte Hashfunktion $S \rightarrow \{0, \ldots, m-1\}$

    Und man kann sie in $O(|S|)$ Zeit finden
  - Der Beweis würde hier zu weit führen, siehe:

    Storing a Sparse Table with O(1) Worst Case Access Time
    Fredman, Komlós, Szemerédi
    Journal of the ACM, Vol 31, No 3, 1984

Histogramme 1/3

$128 \cdot 127$
$= 16.384 - 128$
$= 16.256$

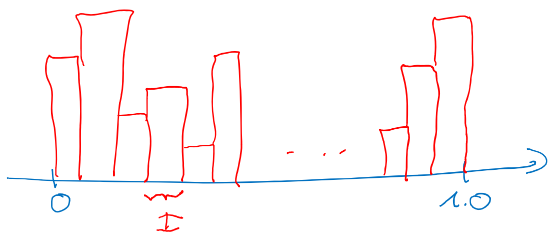
- Brauchen Sie für das Übungsblatt
  - Für vier von den fünf Klassen (Folie 6 – 12), sollen Sie die Kollisions-Wahrscheinlichkeiten von allen x ≠ y berechnen
  - Die Anzahl Paare x, y mit x ≠ y ist u · (u – 1) ... also groß!
  - Die visualisiert man am besten mit einem Histogramm:

    Werte $\overbrace{x_1, x_2, x_3, x_4, \ldots}^{16.256}$ Wertebereich hier [0,1] (es sind W-keiten)

    Unterteile Wertebereich in n disjunkte Teil-Intervalle

    In unserem Fall hier kann man die gleich groß wählen

    Zähle für jedes Teil-Intervall I die Anzahl aller $x_i \in I$



Histogramme 2/3

- Erstellen der Daten
  - Zeilenbasiert in eine Datei ausgeben

0.0	120
0.1	47
0.2	88
…	

oder "Bin"

Erste Spalte:	"Bucket"	(x-Achse)
Zweite Spalte:	Anzahl	(y-Achse)

Histogramme 3/3

- Wie malt man so ein Histogramm?
  - Zum Beispiel mit **gnuplot**

set term pdf	Ausgabe als PDF
set output "data.pdf"	Ausgabedatei
set style fill solid 0.4	Gefüllte Boxen
plot [] [0:200] "data.txt" with boxes	Malen
plot … lt rgb "orange"	Andere Farbe

  - Das geht aber auch mit vielen anderen Programmen, z.B.

    S … oder die open-source Variante R
    Matlab … oder die open-source Variante Octave
    Mathematica
    Excel

Literatur / Links

- Universelle Klassen von Hashfunktionen
  - In Mehlhorn / Sanders:

    4 Hash Tables and Associative Arrays