



Speicherstrukturen

Institut für Datenbanken und Informationssysteme
Prof. Dr. M. Reichert, M. Predeschly, J. Kolb

Besprechung: 4. Juni 12:30 in
Raum o27/121

Übungsblatt 3

Aufgabe 3-1: Dynamisches Spiralhashing

Erstellen Sie die Hash-Verteilung bei einer initialen Dateigröße 2 mit einem Vergrößerungsfaktor 2. Die Hashfunktion $h(pk)$ liefert Werte im Bereich $[0..1)$ zurück. Im Falle, dass die Belegung einer Seite über 50% liegt, vergrößern Sie die Hash-Datei und berechnen die Werteverteilung neu. Führen sie die Operationen Schritt für Schritt aus – Ziel soll es sein das Verfahren von Hand an einem Beispiel durchzurechnen.

Aufgabe 3-2: Nicht-numerische Schlüssel

Gegeben sei der nicht-numerische Schlüssel `www.ulm.de`. Errechnen Sie den für das Hashing benötigten numerischen Wert...

1. ...als bijektive Abbildung mit vollem Zeichensatz.
2. ...als bijektive Abbildung mit reduziertem Zeichensatz `['A'...'Z']`.

Aufgabe 3-3: Signaturverfahren

Implementieren Sie eine einfache Variante des Signaturverfahrens (siehe Skript 4.2.1) in Java.

1. Implementieren Sie zunächst eine Methode der neuen Klasse `Signatur`, die zu einem Dokument (d.h. zu einem String) die Signatur wie folgt bestimmt:
 - Das Dokument wird in Worte (Folgen von Buchstaben) zerlegt.
 - Worte mit weniger als drei Buchstaben werden ignoriert.
 - Worte mit drei oder mehr Buchstaben werden in Fragmente der Länge drei zerlegt, wobei Groß-/Kleinschreibung ignoriert wird.
Beispiel: `Hallo` wird in die Fragmente `hal`, `all`, und `llo` zerlegt.
 - Jedes Fragment wird als dreistellige Zahl zur Basis 26 interpretiert und per Modulo-Funktion auf ein Signaturbit abgebildet.
Beispiel: `hal` entspricht der Zahl `'7 | 0 | 11'` = $7 * 26^2 + 0 * 26 + 11 = 4743$, die auf das Bit $4743 \bmod N$ abgebildet wird, wenn N die Größe einer Signatur in Bits bezeichnet.
 - Der Methode wird ein String und der Wert N übergeben, als Ergebnis wird die Signatur als Objekt der Klasse `java.util.BitSet` zurückgegeben.

2. Erstellen Sie nun eine Methode, die ein Dokument entgegennimmt, dessen Signatur bestimmt (mit Hilfe von 1.)), und dann beide Informationen (als Array, Vector oder eigene Klasse zusammengefasst) in einem `Vector` (Attribut der Klasse) speichert. Dieser Vector soll also die Signaturen und Texte aller übergebenen Dokumente aufsammeln.
3. Eine Anfrage an die gespeicherte Dokumentmenge besteht aus einer Menge von Worten, die ebenfalls als String übergeben wird. Als Resultat sollen alle Dokumente angezeigt werden, die diese Worte enthalten, wobei Groß-/Kleinschreibung wieder ignoriert wird. Implementieren Sie eine entsprechende Anfragefunktion!