6. Übungsblatt

Besprechung: ab 26.06.

Aufgabe 1: WAH Kompression

a) Kodierung

Gegeben sei eine Folge von 128 Bits:

$$B = 1 \times 0; 20 \times 1; 3 \times 0; 79 \times 1; 25 \times 0 \quad (\text{bzw. } 0\underbrace{1\cdots1}_{20\times}000\underbrace{1\cdots1}_{79\times}\underbrace{0\cdots0}_{25\times}).$$

Geben Sie die *WAH-Kodierung* von B an. Nehmen Sie dabei eine **Wortlänge von 32 Bit** an.

b) Logische Operationen

Gegeben seien die WAH-kodierten Darstellungen von zwei Bitfolgen B_X und B_Y[1]

$$\begin{aligned} B_{X,WAH} &= |010\underbrace{1\cdots1}_{28\times}0|000\underbrace{1\cdots1}_{29\times}| \\ B_{Y,WAH} &= |11\underbrace{0\cdots0}_{28\times}10| \ . \end{aligned}$$

Zeigen Sie, wie die Konjunktion

$$B_X \wedge B_Y$$

auf Basis von $B_{X,WAH}$ und $B_{Y,WAH}$ berechnet werden kann, möglichst **ohne** $B_{X,WAH}$ **und** $B_{Y,WAH}$ **zu dekodieren**. Benutzen Sie dazu das Verfahren aus Übungsprojekt 3 basierend auf Artikel [1].

Aufgabe 2: Bloom Filter

Bei der Berechnung von *Joins* ist das *Testen auf Enthaltensein* eine wichtige Operation. Kann ausgeschlossen werden, dass der Joinschlüssel eines Tupels in der Join-Tabelle enthalten ist, kann das Tupel verworfen werden.

[1] Zur besseren Lesbarkeit wurden Word-Grenzen durch '|' markiert.

Der *Bloom Filter* ist eine kompakte probabilistische Datenstruktur, die das Testen auf Enthaltensein ermöglicht. Der Filter besteht aus einem m-dimensionalen Bitvektor and k Hashfunktionen, die Werte auf eine Zahl von 0 bis $m-1$ abbilden. Jeder Wert der Menge $\mathcal{M}$ wird mit jeder der k Funktionen gehasht und für jeden Hash-Wert wird ein Bit an der entsprechenden Position des Bitvektors gesetzt.

Für die Testmenge $\mathcal{T}$ werden Anfragen an den Filter gestellt. Jeder Wert $t \in \mathcal{T}$ wird mit jeder der k Funktionen gehasht. Steht an mindestens einer der Positionen im Bitvektor eine 0, kann ausgeschlossen werden dass t in $\mathcal{M}$ enthalten ist.

a) Beispiel

Veranschaulichen Sie zunächst die Funktionsweise des Bloom Filters mit zwei einfachen Hash-Funktionen und zwei kleinen **Beispielmengen** $\mathcal{M}$ und $\mathcal{T}$ von ganzen Zahlen. Als Hash-Funktionen eignen sich Beispielweise eine Kombination aus Modulo und Quersumme. Achten Sie darauf, dass die Hash-Werte im Bereich 0 und $m-1$ liegen müssen.

b) Anfrage

Skizzieren Sie die Verwendung eines Bloom Filters für die folgende Starjoin-Anfrage:

```
select sum(f.SalesAmount), dt.Region, dd.MonthName
  from FactSales f, DimTerritory dt, DimDate dd
 where f.TerritoryKey = dt.TerritoryKey
   and f.OrderDateKey = dd.DateKey
   and dt.TerritoryCountry = 'United States'
   and dd.CalendarYear = 2008
group by dt.Region, dd.MonthName
```

c) Fehlerwahrscheinlichkeit

Der Bloom Filter kann falsche positive Aussagen machen. Das heißt, dass der Bloom Filter für ein Tupel $t \in \mathcal{T}$ eine positive Antwort gibt, obwohl $t \notin \mathcal{M}$. Die Wahrscheinlichkeit dafür ist

$$p = (1 - (1 - \frac{1}{m})^{k \cdot n})^k$$

wobei m die Länge des Bitvektors, k die Zahl der Hashfunktionen und $n = \mid \mathcal{M} \mid$ ist. Erklären Sie, warum die Formel diese Fehlerwahrscheinlichkeit berechnet. Bestimmen Sie die Mindestlänge des Bitvektors für $k = 7$, $n = 5000$ unter der Bedingung $p \leq 0.01$.

Literatur

[1] Kesheng Wu, Ekow J. Otoo und Arie Shoshani: Optimizing bitmap indices with efficient compression. ACM Trans. Database Syst, Band 31, Nummer 1. 2006.