Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen
Prof. Dr. Ernst W. Mayr/Dr. Sven Kosub
Stefan Eckhardt

Wintersemester 2004/05
Übungsblatt 5
20. Dezember 2004

# Effiziente Algorithmen und Datenstrukturen I

*Abgabetermin: 29.11.2004* **vor** *der Vorlesung*

Beim quadratischen Sondieren verwenden wir als erweiterte Hashfunktion eine Funktion der Form

$$\bar{h}(k,i) =_{\text{def}} (h(k) + c_1 \cdot i + c_2 \cdot i^2) \bmod n$$

wobei $h : U \to \{0, 1, \ldots, h-1\}$ für endliches Universum $U$ und $c_1, c_2 \in \mathbb{N}$ sind. Betrachten Sie folgenden Algorithmus auf Eingabe $k \in U$ und Array $A = [0..n-1]$:

(1) $j := 0$
(2) $i := h(k)$
(3) **while** $A[i] \neq k$ **of** $A[i]$ leer
(4) $\quad j := (j+1) \bmod n$
(4) $\quad i := (i+j) \bmod n$
(4) **return** „Suche beendet“

Nehmen Sie $n$ als Zweierpotenz an.

(a) Zeigen Sie, dass dieser Algorithmus Quadratisches Sondieren realisiert. Geben Sie dazu geeignete Konstanten $c_1$ und $c_2$ an.

(b) Zeigen Sie, dass dieser Algorithmus im schlechtesten Fall alle Array-Positionen durchläuft.

## Aufgabe 1

Eine Familie $\mathcal{H}$ von Hashfunktionen von einem Universum $U$ in die Menge $B$ heißt $\varepsilon$-universell, falls für alle Paare $k, l \in U$ mit $k \neq l$ gilt

$$\Pr[h(k) = h(l)] \leq \varepsilon,$$

wobei $h$ eine gleichverteilt aus $\mathcal{H}$ gezogene Hashfunktion sei. Zeigen Sie, dass für eine $\varepsilon$-universelle Familie von Hashfunktionen gelten muß

$$\varepsilon \geq \frac{1}{|B|} - \frac{1}{|U|}.$$

## Aufgabe 2

Man kann $n$ Schlüssel dadurch sortieren, dass man zuerst einen binären Suchbaum $T$ initialisiert, danach alle $n$ Schlüssel per $\textsc{Insert}(T, k)$ in den Suchbaum $T$ einfügt und schließlich alle Schlüssel in $T$ in ihrer *inorder*-Reihenfolge ausgibt. Untersuchen Sie die worst-case und best-case Laufzeiten dieses Sortierverfahrens.

## Aufgabe 3

Die Prozedur $\textsc{TreeDelete}(T, z)$ löscht einen Knoten $z$ aus dem binären Suchbaum $T$. Falls $z$ zwei Kinder hat, wird dazu ein geeigneter Nachfahre von $z$ bestimmt, der gelöscht wird und dessen Schlüssel in den Knoten $z$ kopiert wird. Eine „fairere “Strategie wäre, nicht nur unter den Nachfahren einen geeigneten Kandidaten zu suchen, sondern auch die Vorfahren von $z$ zu betrachten. Zeigen Sie, wie so eine fairere Strategie implementiert werden könnte?