Technische Universität Berlin **SS 2013**
Fakultät II, Institut für Mathematik
Sekretariat MA 5–2, Frau Kiefer

Prof. Dr. Martin Skutella
Torsten Gellert
Martin Groß
Dr. Max Klimm

Aylin Acikel, Katharina Bütow, Christian Döblin, Alexander Hopp, Daniel Kuske, Olivia Röhrig, Robert Rudow, Daniel Schmand, Hendrik Schrezenmaier, Mona Setje-Eilers, Judith Simon, Sebastian Spies, Fabian Wegscheider, Jan Zur

9. Übungsblatt Computerorientierte Mathematik II

Abgabe: 11.06.2013 (bis 14:15 Uhr in MA 004)

1. Aufgabe (3 + 6 Punkte)

(a) Betrachtet eine Hash-Tabelle der Größe $m = 9$. Als Hashfunktion wird die Multiplikationsmethode mit $A = \frac{5}{11}$ und als Kollisionsbehandlung *chaining* gewählt.

  (i) Fügt die Schlüsselwerte 77, 13, 54, 8, 23, 9, 12, 48, 20, 3, 97 in dieser Reihenfolge in eine Hashtabelle ein und gebt die endgültige Belegung der Tabelle an.

  (ii) Was passiert, wenn sehr viele gleichverteilte Schlüssel mit dieser Hashfunktion in diese Tabelle eingefügt werden? Begründet eure Antwort!

(b) Die Daten 15, 3, 31, 41, 19, 4, 29, 45, 14, 2, 48, 36, 16, 65, 1 sollen in dieser Reihenfolge in eine Hashtabelle der Länge m eingefügt werden, die *double hashing* als Kollisionsbehandlung verwendet. Die primäre Hashfunktion ist in allen Fällen $h(x) = x \bmod m$, die sekundäre $g(x) = 1 + (x \bmod q)$.

Zeichnet in den folgenden drei Fällen die Tabelle geeignet auf, zählt beim Einfügen die Anzahl (ggf. wiederholter) Kollisionen und vergleicht diese Zahlen. Welche typischen Effekte im Zusammenspiel der beiden Hashfunktionen könnt ihr an diesen Beispielen beobachten? Gebt eine kurze Begründung!

  (i) $m = 16,\ q = 9$

  (ii) $m = 17,\ q = 1$ *(linear probing)*

  (iii) $m = 17,\ q = 10$.

2. Aufgabe (1 + 3 + 5 + 2 Punkte)

Es sollen $n = 2^m - 1$, $m \in \mathbb{N}$ vorher bekannte Datensätze in einer Datenstruktur D gehalten und danach möglichst schnell gesucht werden können. Dabei ist bekannt, dass ein Anteil $r \cdot n$ der Daten, $r \in (0, 1)$ insgesamt mit der Wahrscheinlichkeit $p \in [\frac{1}{2}, 1]$ angefragt wird, der restliche Anteil $(1 - r)n$ mit der Wahrscheinlichkeit $1 - p$. Beim Einfügen in die Datenstruktur ist bekannt, ob der Datensatz zum Anteil $r \cdot n$ gehört, nicht jedoch beim Suchen.

Ein möglicher Ansatz ist es, die Datenstruktur D in zwei gleichartig funktionierende Teile aufzuteilen, einen Teil D_r für die häufig nachgefragten Daten und einen zweiten Teil D' für den Rest. Man sucht dann stets zuerst in D_r. Nur bei erfolgloser Suche würde dann noch in D' gesucht.

In dieser Aufgabe soll untersucht werden, bei welchen Datenstrukturen und Wahrscheinlichkeitsverhältnissen p und Anteilsverhältnissen r sich diese Aufspaltung gegenüber einer einzigen „reinen" Datenstruktur D lohnen würde. Dabei soll der *average case* untersucht werden.

(a) Sei D eine allgemeine Datenstruktur und sei $a(n)$ der mittlere Aufwand für eine erfolgreiche Suche auf einer Datenstruktur, die mit n Elementen gefüllt ist und $b(n)$ sei der mittlere Aufwand für eine erfolglose Suche. Interpretiere die folgende Ungleichung. Wann gilt sie?

$$p \cdot a(\,rn\,) + (1-p) \cdot \Big(b(\,rn\,) + a\big(\,(1-r)n\,\big)\Big) \leq a(n)$$

(b) Der Typ der Datenstruktur D sei eine lineare (unsortierte) Liste. Gebt $a(n)$ und $b(n)$ an und beweist, für welche Werte für p und r sich eine Aufspaltung lohnt.

(c) Der Typ von D sei ein bis in die letzte Schicht voller binärer Suchbaum (d. h. der Binärbaum hat maximale Anzahl an Knoten zur Höhe h).

(i) Zeigt, dass $a(n) = \log_2 n - 1 + o(1)$ und $b(n) = \log_2 n + o(1)$.

(Tipp: Ihr könnt dazu die Identität $\sum_{k=1}^{m} k2^{k-1} = (m-1)2^m + 1$ *verwenden.)*

(ii) Zeigt, dass sich die Aufspaltung nur lohnt, wenn gilt:

$$p \geq \frac{\log_2\big(r(1-r)n\big)}{\log_2\big((1-r)n\big)}.$$

Rechnet dafür mit den Näherungen $a(n) = \log_2 n - 1$ und $b(n) = \log_2 n$.

(d) Der Typ von D sei nun eine Hashtabelle geeigneter Größe. Wann lohnt sich eine Aufspaltung?

Viel Spaß und Erfolg!