Arbeitsblatt: Open Hashing

Aufgabe 1 Clustering

Sei n die Tabellengrösse und k die Grösse eines Clusters in der Tabelle. Beantworten Sie folgende Fragen zu linearem Sondieren:

a) Wie hoch ist die Wahrscheinlichkeit, dass beim nächsten Einfügen eines neuen Elements der Cluster wächst?

(k+2)/n. Zeigt der Hashwert auf eine Stelle im Cluster oder auf eine direkt angrenzende Stelle, wächst der Cluster beim Einfügen.

b) Wie hoch ist die Wahrscheinlichkeit, dass das Einfügen eines neuen Elements mindestens 3 Sondierungs-Schritte benötigt (d.h. dreimal ist das Feld bereits besetzt)?

(k-2)/n. Der Hashwert muss auf eine der ersten k-2 Stellen im Cluster zeigen.

c) Nehmen Sie nun an, die Tabelle hat Grösse 23 und drei Cluster der Grösse 5. Wie viele Sondierungs-Schritte erwarten Sie im Schnitt beim Einfügen eines neuen Elements?

Es gibt 23 – 15 = 8 freie Stellen, wo nicht sondiert werden muss.

Ausserdem gibt es je drei Stellen, wo 5 Mal (erste Position in einem Cluster), 4 Mal, 3 Mal, 2 Mal, 1 Mal (letzte Stelle in einem Cluster) sondiert werden muss. Es gilt 3 * (5 + 4 + 3 + 2 + 1) = 45.

Wenn wir annehmen, dass der Hashwert mit gleicher Wahrscheinlichkeit auf eine Stelle zeigt, dann ist die erwartete Anzahl Sondierungen 45 / 23 = 1.96, also in etwa 2.

Aufgabe 2 Step

Double Hashing ist besser als Lineares Sondieren, weil die Sondierungsschrittweite für verschiedene Elemente verschieden gewählt wird. Warum genügt es nicht, einfach z.B. step = 7 für alle Elemente zu wählen? Begründen Sie Ihre Antwort mit einem Beispiel.

Wir bringen dadurch das clustering nicht wirklich los, es ist graphisch einfach nicht mehr so gut zu sehen: würden wir aber jeweils die Positionen mit Abstand 7 nebeneinander betrachten, dann zeigte sich das gleiche Bild.

z.B. Tabelle mit Grösse 9, Hashwerte 1, 17, 6, 10, 8. Die Hashwerte würden alle auf den Positionen 1, 8 und 6 abgebildet werden. Diese liegen in der Sondierreihenfolge direkt «nebeneinander».

	1	8		10		6		17
0	1	2	3	4	5	6	7	8

				1	17	6	10	8
0	7	5	3	1	8	6	4	2

Aufgabe 3 Load Factor

Sei L der Load Factor der Hashtabelle. Bantworten Sie folgende Fragen zu Separate Chaining. Nehmen Sie an, dass die Schlüssel zufällig verteilt sind.

a) Was ist die erwartete Länge einer Liste in der Tabelle?

Die erwartete Länge ist genau L, weil die Summe aller Listenlängen genau der Summe aller Elementen in der Hashtabelle entspricht.

b) Wie hoch ist der erwartete Aufwand einer erfolgreichen Suche (d.h. das Element ist in der Hashtabelle enthalten)?

O(1 + L/2). Im Erwartungswert hat man das Element nach Absuchen der halben Liste gefungen.

c) Wie hoch ist der erwartete Aufwand einer erfolglosen Suche?

O(1 + L). Man muss die ganze Liste durchlaufen.

Aufgabe 4 Hands-On

a) In eine leere Hashtabelle der Grösse 11, fügen Sie die untenstehenden Schlüssel ein.

31, 45, 15, 99, 11, 5, 47, 4, 14, 28

Benutzen Sie dafür einmal lineares Sondieren mit H(x) = x % 11 und einmal Double Hashing mit zweiter Hashfunktion H2(x) = 1 + x % 9.

lineares Sondieren:

99	45	11	47	15	5	4	14	28	31	
0	1	2	3	4	5	6	7	8	9	10

double Hashing:

99	45		11	15	5	47	28	4	31	14
0	1	2	3	4	5	6	7	8	9	10

b) Löschen Sie nun aus beiden Tabellen die Schlüssel 99 und 11 und fügen Sie danach den Schlüssel 23 ein.

lineares Sondieren:

-	45	23	47	15	5	4	14	28	31	
0	1	2	3	4	5	6	7	8	9	10

double Hashing:

-	45	23	-	15	5	47	28	4	31	14
0	1	2	3	4	5	6	7	8	9	10