
Grundlagen: Algorithmen und Datenstrukturen

Abgabetermin: In der jeweiligen Tutorübung

Hausaufgabe 1

Implementieren Sie eine Warteschlange auf der Basis eines Dynamischen Arrays.
Erweitern Sie die abstrakte Klasse `UQueue` zu einer Klasse `UIQueue`.
Verwenden Sie die auf der Übungsseite bereitgestellte Klasse `UIsArray` und modifizieren Sie diese *nicht*.

Achten Sie bei der Abgabe Ihrer Aufgabe darauf, dass Ihre Klasse `UIQueue` heißt und auf den Rechnern der Rayhalle (rayhalle.informatik.tu-muenchen.de) mit der bereitgestellten Datei `main_q` kompiliert werden kann. Anderenfalls kann eine Korrektur nicht garantiert werden. Achten Sie darauf, dass Ihr Quelltext ausreichend kommentiert ist. Schicken Sie die Lösung per Email mit dem Betreff `[GAD] Gruppe <Gruppennummer>` an ihren Tutor.

Lösungsvorschlag

Siehe Übungswebseite

Hausaufgabe 2

Bestimmen Sie die Wahrscheinlichkeit, dass von n Personen mindestens zwei Personen am gleichen Tag im Jahr Geburtstag haben.

Erläutern Sie den Zusammenhang zwischen Hashing und dem eben gezeigten Geburtstagsparadoxon.

Hinweis: Bestimmen Sie die Wahrscheinlichkeit, des Komplementär-Ereignisses: Keine zwei Person haben am gleichen Tag Geburtstag. Ein Jahr hat hier immer 365 Tage.

Lösungsvorschlag

Das Ereignis, mindestens zwei Personen haben an einem Tag Geburtstag setzt sich aus vielen Ereignissen zusammen: $\forall 2 \leq i \leq n$: Es gibt einen Tag x so dass genau i Personen an dem Tag x Geburtstag haben. Allerdings gibt es nur ein Ereignis welches nicht in dieser Menge von Ereignissen eingeschlossen ist: Alle n Personen haben an unterschiedlichen Tagen Geburtstag. Daher betrachten wir die Wahrscheinlichkeit x für das letztere Ereignis und geben die Wahrscheinlichkeit y für das erste Ereignis mit $y = 1 - x$ an.

Die Wahrscheinlichkeit, dass keine zwei Personen am gleichen Tag Geburtstag haben ist die Anzahl aller Möglichkeiten, dass alle n Personen an unterschiedlichen Tagen Geburtstag haben, dividiert durch die Anzahl aller möglichen Geburtstage der n Personen.

Wir bestimmen zuerst die Anzahl aller Geburtstage von n Personen. Diese ist gerade 365^n (für jede Person gibt es 365 mögliche Geburtstage; das ganze n -Mal).

Die Anzahl aller Geburtstage von n Personen die unterschiedlich sind ist 0 wenn $n > 365$. Sonst gilt: Die erste Person belegt einen der 365 Tage. Die zweite Person kann nur noch an einem der restlichen 364 Tage Geburtstag haben usw.. Insgesamt ergibt sich also 365^n . Die Wahrscheinlichkeit, dass es unter n Personen mindestens zwei Personen gibt, die am selben Tag Geburtstag haben ist also

$$1 - \frac{365^n}{365^n}.$$

Beachtlich ist nun die Auswertung dieses Ausdrucks:

n	Wahrscheinlichkeit	n	Wahrscheinlichkeit	n	Wahrscheinlichkeit
2	0.00273973	22	0.475695	42	0.91403
3	0.00820417	23	0.507297	43	0.923923
4	0.0163559	24	0.538344	44	0.932885
5	0.0271356	25	0.5687	45	0.940976
6	0.0404625	26	0.598241	46	0.948253
7	0.0562357	27	0.626859	47	0.954774
8	0.0743353	28	0.654461	48	0.960598
9	0.0946238	29	0.680969	49	0.96578
10	0.116948	30	0.706316	50	0.970374
11	0.141141	31	0.730455	51	0.974432
12	0.167025	32	0.753348	52	0.978005
13	0.19441	33	0.774972	53	0.981138
14	0.223103	34	0.795317	54	0.983877
15	0.252901	35	0.814383	55	0.986262
16	0.283604	36	0.832182	56	0.988332
17	0.315008	37	0.848734	57	0.990122
18	0.346911	38	0.864068	58	0.991665
19	0.379119	39	0.87822	59	0.992989
20	0.411438	40	0.891232	60	0.994123
21	0.443688	41	0.903152		

Es fällt auf, dass recht wenige Personen benötigt werden, um eine hohe Wahrscheinlichkeit zu erhalten, dass mindestens 2 von diesen Personen an einem Tag Geburtstag haben.

Der Zusammenhang zum Hashing ist der folgende: Den Geburtstag kann man als Hashfunktion für eine Person betrachten. Haben nun zwei Personen an dem selben Tag Geburtstag so hat man eine Kollision und man muss sich Gedanken machen, was man mit dieser Kollision macht (Chaining, Linear Probing, Double Hashing, Cuckoo Hashing, ...). Das Geburtstagsparadoxon zeigt nun sehr eindringlich (dies ist trotzdem kein Beweis in voller Allgemeinheit), dass solche Kollisionen, auch bei einer verhältnismäßig großen Hashtabelle (vgl. $n = 23$, $m = 365$) definitiv keine Seltenheit sind.

Aufgabe 1

Veranschaulichen Sie Hashing mit Chaining und Hashing mit Linear Probing. Fügen Sie dazu die Elemente

3, 11, 9, 7, 14, 23, 4, 12, 8, 15, 0

in eine Hash-Tabelle der Größe 11 ein.

Benutzen Sie die Hashfunktion

$$h_5(x) = 5x \text{ mod } 11.$$

Lösungsvorschlag

Hashing mit Chaining:

Insert(3):

0	1	2	3	4	5	6	7	8	9	10
			3							

Insert(11):

0	1	2	3	4	5	6	7	8	9	10
11			3							

Insert(9):

0	1	2	3	4	5	6	7	8	9	10
11	9		3							

Insert(7):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3						

Insert(14):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3						
				14						

Insert(23):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3	23					
				14						

Insert(4):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3	23				4	
				14						

Insert(12):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3	23				4	
				14	12					

Insert(8):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3	23		8		4	
				14	12					

Insert(15):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3	23		8		4	
				14	12					

Insert(0):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3	23		8		4	
0				14	12					

Hashing mit Linear Probing:

Insert(3):

0	1	2	3	4	5	6	7	8	9	10
			3							

Insert(11):

0	1	2	3	4	5	6	7	8	9	10
11			3							

Insert(9):

0	1	2	3	4	5	6	7	8	9	10
11	9		3							

Insert(7):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3						

Insert(14):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3	14					

Insert(23):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3	14	23				

Insert(4):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3	14	23			4	

Insert(12):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3	14	23	12		4	

Insert(8):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3	14	23	12	8	4	

Insert(15):

0	1	2	3	4	5	6	7	8	9	10
11	9	7		3	14	23	12	8	4	15

Insert(0):

0	1	2	3	4	5	6	7	8	9	10
11	9	7	0	3	14	23	12	8	4	15

Aufgabe 2 (10 Punkte)

Die Werte {Apfel, Banane, Kirsche, Himbeere, Melone} sollen in einer Hashtabelle der Größe $n = 4$ untergebracht werden. Es seien folgende Hashfunktionen gegeben:

f_1 : Apfel \mapsto 4	Banane \mapsto 2	Kirsche \mapsto 2	Himbeere \mapsto 1	Melone \mapsto 4
f_2 : Apfel \mapsto 3	Banane \mapsto 4	Kirsche \mapsto 2	Himbeere \mapsto 3	Melone \mapsto 4
f_3 : Apfel \mapsto 2	Banane \mapsto 2	Kirsche \mapsto 4	Himbeere \mapsto 1	Melone \mapsto 1
f_4 : Apfel \mapsto 1	Banane \mapsto 3	Kirsche \mapsto 3	Himbeere \mapsto 4	Melone \mapsto 4
g_1 : Apfel \mapsto 1	Banane \mapsto 1	Kirsche \mapsto 3	Himbeere \mapsto 2	Melone \mapsto 3
g_2 : Apfel \mapsto 2	Banane \mapsto 4	Kirsche \mapsto 2	Himbeere \mapsto 3	Melone \mapsto 4
g_3 : Apfel \mapsto 4	Banane \mapsto 4	Kirsche \mapsto 1	Himbeere \mapsto 4	Melone \mapsto 2
g_4 : Apfel \mapsto 3	Banane \mapsto 1	Kirsche \mapsto 2	Himbeere \mapsto 3	Melone \mapsto 3
g_5 : Apfel \mapsto 4	Banane \mapsto 2	Kirsche \mapsto 2	Himbeere \mapsto 2	Melone \mapsto 3

In der Vorlesung haben wir den Begriff der c -universellen Hashfunktionen kennengelernt.

- (a) Geben Sie für die Familie $\mathcal{H}_1 = \{f_1, f_2, f_3, f_4\}$ das kleinste c an, so dass \mathcal{H}_1 c -universell ist.
- (b) Finden Sie eine möglichst kleine Familie $\mathcal{H}_2 \subseteq \{g_1, g_2, g_3, g_4, g_5\}$, die 1-universell ist.

Begründen Sie Ihre Aussagen.

Lösungsvorschlag

Eine Familie \mathcal{H} von Hashfunktionen ist nach Definition c -universell, wenn für *alle* Paare x, y mit $x \neq y$ gilt

$$\frac{|\{f \in \mathcal{H} : f(x) = f(y)\}|}{|\mathcal{H}|} \leq \frac{c}{n}.$$

Für $n = 4$ und $|\mathcal{H}_1| = 4$ bestimmt die Anzahl der Kollisionen pro Paar (x, y) direkt das c . Wir stellen zunächst eine Tabelle der Paare und ihren Kollisionen auf:

Paar	f_1	f_2	f_3	f_4	g_1	g_2	g_3	g_4	g_5
Apfel/Banane			x		x		x		
Apfel/Kirsche						x			
Apfel/Himbeere		x					x	x	
Apfel/Melone	x							x	
Banane/Kirsche	x			x					x
Banane/Himbeere							x		x
Banane/Melone		x				x			
Kirsche/Himbeere									x
Kirsche/Melone					x				
Himbeere/Melone			x	x				x	

Da es für \mathcal{H}_1 für die Werte Banane/Kirsche sowie Himbeere/Melone jeweils zwei Kollisionen gibt und sonst zwischen alle Wertepaaren höchstens eine Kollision auftritt, ist \mathcal{H}_1 also 2-universell.

Da \mathcal{H}_2 1-universell sein soll und jede Funktion aufgrund des Schubfachprinzips eine Kollision verursachen muss, muss also \mathcal{H}_2 mindestens 4 Funktionen beinhalten.

Definieren wir nun $\mathcal{H}_2 := \{g_1, g_2, g_4, g_5\}$ so ist diese Familie 1-universell, da für jedes Paar nur eine Kollision auftritt.