Karlsruher Institut für Technologie

Institut für Theoretische Informatik
T.T.-Prof. Dr. Thomas Bläsius
https://scale.iti.kit.edu

Übungsblatt 14

Algorithmen I – Sommersemester 2023

Abgabe im ILIAS bis 04.08.2023, 18:00 Uhr

Die Abgabe erfolgt als *eine* PDF-Datei über das Übungsmodul in der Gruppe deines Tutoriums im ILIAS.
Beachte bitte die Hinweise zum Bearbeiten auf der Webseite.

Deine Abgabe zu diesem Blatt wird nur korrigiert, wenn du bisher mindestens 110 und echt weniger als 130 Punkte hast. Schreib daher bitte deine aktuelle Punktzahl auf die Abgabe. Die Aufgaben auf diesem Blatt geben ausschließlich Bonuspunkte.

Aufgabe 1 - Arbeitsschutz geht alle an! (0+5 Punkte)

Der Damm nördlich der Alpen wächst rasant. Jean-Claude und Justin kommen schnell voran, scheinen dabei aber etwas schludrig zu sein. Der Bauaufsichtsbiberin Bibi ist aufgefallen, dass Stämme aus entfernten Schichten einfach den Bach runter gehen. Im BGB (biberliches Gesetzbuch) ist in §*B1 b)-e) RN:23* festgelegt, dass von einem Damm abgebautes Holz vor der endgültigen Entsorgung *sortiert* gelagert werden muss. Bibi erinnert sich an das zweite Übungsblatt, wo Stämme mit einer *Stabilitätszahl* versehen wurden und verlangt, dass die abgebauten Stämme bezüglich dieser Zahl sortiert werden.

Die Datenstruktur die zur Verwaltung des Dammbaus verwendet wird, verfügt dann über die folgenden Operationen.

- `build`: Fügt in Zeit $\Theta(1)$ eine neue Schicht zum Damm hinzu.
- `dismantle`(k): Entfernt die obersten $k > 0$ Schichten vom Damm, sortiert sie, und fügt sie ins Lager ein. Dafür wird Zeit $\Theta(k \log(k))$ benötigt.
- `flush`: Spült die m aktuell gelagerten Objekte in $\Theta(m)$ Zeit den Bach runter.

Als Justin das hört, fährt er aus dem Fell: “Das kann doch nicht sein! Jetzt muss ich nicht nur die teuren Abbauarbeiten übernehmen, sondern auch noch sortieren?!”

a*) Verdeutliche Justins Beschwerde, indem du für jedes $n \in \mathbb{N}$ eine Sequenz von Operationen angibst, sodass
   - die Sequenz aus $\Theta(n)$ Operationen besteht,
   - es eine `flush`-Operation gibt, die Laufzeit $\Theta(\log(n))$ hat und

- es eine `dismantle`-Operation gibt, die Laufzeit $\Theta(n \log(n))$ hat.

Begründe deine Antwort. (2 Punkte)

Mit deiner Unterstützung fühlt sich Justin bestärkt und legt noch einen drauf: "Und um die Entsorgung muss ich mich bestimmt auch noch selbst kümmern!" Bibi bewahrt die Ruhe und antwortet gelassen: "Keine Sorge. Um die Entsorgung wird sich Biber Dammeron kümmern. Außerdem ist der Arbeitsaufwand für jeden Handgriff am Ende ja quasi doch nur logarithmisch."

b*) Hilf Bibi, indem du zeigst, dass in jeder Abfolge n Operationen, beliebig zusammengesetzt aus `build-`, `dismantle` und `flush`-Operationen, jede einzelne nur amortisiert Laufzeit $O(\log(n))$ hat.
Hinweis: Verwende dazu eine der vorgestellten Methoden für amortisierte Analyse aus der Vorlesung. (3 Punkte)

Aufgabe 2 - Laufzeit gesucht (0+3 Punkte)

Was ist die Laufzeit eines möglichst effizienten Algorithmus, der das Gewünschte umsetzt? Gib die Laufzeit jeweils O-Notation an.

a*) Die Reihenfolge der Elemente in einer einfach verketteten Liste mit n Elementen soll invertiert werden, sodass das letzte Elemente an erster Stelle steht, das vorletzte an zweiter, usw. (1 Punkt)

b*) Der Durchmesser (maximale Distanz über alle Knotenpaare) in einem gewichteten Graphen mit n Knoten soll bestimmt werden. (1 Punkt)

c*) In einem sortierten Array der Größe n soll das $\log(n)$-kleinste Element bestimmt werden. (1 Punkt)

Aufgabe 3 - Datenstruktur gesucht (0+3 Punkte)

Bestimme für folgende Situationen, welche möglichst einfache Datenstruktur jeweils geeignet ist und gib an, welche Operationen der von dir gewählten Datenstruktur dabei relevant sind.
Hinweis: In der Vorlesung haben wir folgende Datenstrukturen kennengelernt: Listen, (dynamische) Arrays, Hashtabellen, binäre Heaps, $(2, 3)$-Bäume, Union-Find, Adjazenzliste.

a*) Für ein Videospiel werden im Laufe der Zeit Level entworfen, denen von Playtestern ein Spannungswert zwischen 0 und 100 zugewiesen wird. Um die Spannung im Spiel ausgeglichen zu halten, soll zu jedem Zeitpunkt festgestellt werden können, ob man die Level so in Paare zusammenfassen kann, dass sich die beiden Spannungswerte zu 100 aufsummieren. (1 Punkt)

b*) Bei einem Wettkampf ist jeder teilnehmenden Person eine Zahl zugewiesen, die ihre Stärke repräsentiert. In der Disziplin *Dynamisches Tauziehen* kann ein Team unterstützt werden, indem sich eine Person auf der zugehörigen Seite einfach hinten einreiht und mit zieht. Das Team verlassen darf nur, wer gerade ganz Außen steht. Nun soll immer bestimmt werden, welches Team gerade die Nase vorn hat, was dadurch definiert ist, bei wem die ganz außen stehende Person die stärkere ist. (1 Punkt)

c*) Über mehrere Tage bereiten sich Leute ununterbrochen auf eine Klausur vor. Zu den unterschiedlichsten Zeiten fangen sie mit dem Lernen an. Sobald eine Person mit ihrem Latein am Ende ist, schließt sie sich einer anderen Person oder Gruppe an. Auch Gruppen schließen sich anderen Gruppen an, wenn sie nicht weiter kommen. Am Ende versammeln sich alle Gruppen bei der Klausur. Zu jedem Zeitpunkt soll man für eine gegebene Gruppe sagen können, welche die Person war, die am zeitigsten mit dem Lernen begonnen hat. (1 Punkt)

Aufgabe 4 - Wahr oder falsch? (0+3 Punkte)

Entscheide für jede der folgenden Aussagen, ob sie wahr oder falsch ist und gib jeweils eine kurze Begründung an.

a*) Sei G ein vollständiger Graph (jedes Knotenpaar ist mit einer Kante verbunden) mit sechs Knoten bei dem die Kanten mit zwei Farben gefärbt sind. Dann existiert ein einfarbiges Dreieck in G. (1 Punkt)

b*) Unter Verwendung einer Hashtabelle kann ich in $O(n)$ Worst-Case Laufzeit herausfinden, ob sich Duplikate in einer Menge von n natürlichen Zahlen befinden. (1 Punkt)

c*) Sei G ein vollständiger Graph (es existiert eine Kante zwischen jedem Paar von Knoten) und T ein DFS-Baum von G. Dann hat jeder Knoten in T höchstens Grad 2. (1 Punkt)

Aufgabe 5 - „Wer abschreibt betrügt sich selbst“ (0+6 Punkte)

Die Tutorin Tabea stellt fest, dass manche der abgegebenen Lösungen für eine besonders schwere Übungsaufgabe verdächtig ähnlich sind. Um dies genauer zu untersuchen, wollen wir für verschiedene Lösungen genau herausfinden, wie ähnlich diese sind. Hierfür nehmen wir an, dass eine Lösung ein String über einem endlichen Alphabet ist und fragen uns für zwei Strings s_1 und s_2, was die geringste Anzahl an Änderungen ist, mit denen man s_1 in s_2 überführen kann. Mögliche Änderungen sind:

- an einer beliebigen Stelle ein neues Zeichen einfügen,
- ein beliebiges Zeichen entfernen oder

- ein beliebiges Zeichen durch ein anderes ersetzen.

Beispielsweise können wir das Wort „Algorithmus“ mit 5 Änderungen in „Altruismus“ überführen:

```
Algor ithmus   (ersetze g durch t)
Altor ithmus   (entferne o)
Alt r ithmus   (füge u ein)
Alt ruithmus   (füge s ein)
Alt ruishmus   (entferne h)
Alt ruis mus
```

Wir sagen dass String s_1 *Distanz* k zu String s_2 hat, wenn mindestens k Änderungen nötig sind, um s_1 in s_2 zu überführen.

a*) Angenommen s_1 hat Distanz k zu s_2. Hat dann auch s_2 Distanz k zu String s_1? Begründe deine Antwort! (1 Punkt)

Wir wollen nun ein dynamisches Programm entwickeln, um für gegebene Strings s_1 und s_2 die Distanz von s_1 zu s_2 zu berechnen. Hierfür wollen wir für jede Länge i eines Präfixes $s_1[1..i]$ und jede Länge j eines Präfixes $s_2[1..j]$ eine Teillösung $X[i, j]$ speichern.

b*) Gib an, welche Bedeutung die Teillösung $X[i, j]$ haben soll und gib darauf aufbauend die Rekurrenz an. (3 Punkte)
Hinweis: achte darauf auch die Basisfälle der Rekurrenz anzugeben

c*) Wir wollen jetzt zusätzlich für jeden Eintrag (i, j) des DPs einen Zeiger speichern, mit dem wir eine kürzeste Folge an Änderungen, welche s_1 in s_2 überführt, rekonstruieren können. Gib an, wie diese Zeiger gesetzt werden müssen und wie man mit ihnen die Änderungen rekonstruieren kann. (2 Punkte)