Internet-Algorithmen

Sommersemester 2014

Prof. Dr. Georg Schnitger
Hannes Seiwert

Arbeitsgruppe Theoretische Informatik, Institut für Informatik

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Übungsblatt 4

Ausgabe: 26.05.2014
Abgabe: 02.06.2014 vor Vorlesungsbeginn

Hinweis: Alle Lösungen sind ordentlich und in lesbarer Schrift zu verfassen. Fasse dich kurz, beschränke deine Erläuterungen und Rechnungen auf die wesentlichen Punkte.

Hinweis: In Aufgabe 2 und 3 sind Algorithmen zu entwerfen. Gehe dabei in drei Schritten vor:
1. Beschreibe kurz die **Idee** deines Algorithmus. Falls nicht offensichtlich, begründe hier, warum dein Verfahren korrekt ist.
2. Gib deinen Algorithmus in **Pseudocode** an.
3. Analysiere den benötigten **Speicherplatz** und die **Laufzeit** deines Algorithmus bzw. begründe, warum die Vorgaben eingehalten werden.

Hinweis: Die Gesamtpunktzahl beträgt 24 Punkte. Darüber hinaus erworbene Punkte werden als Bonuspunkte angerechnet.

Aufgabe 4.1. (3+5)

Zahlenschwund

a) Die Zahlen $1, \ldots, n$ werden in einer vorher nicht bekannten Reihenfolge ausgegeben. Allerdings fehlt genau eine der Zahlen. **Bestimme** die fehlende Zahl mit Speicherplatz $O(\log n)$.

b) Wie in Teil (a) werden die Zahlen $1, \ldots, n$ wieder permutiert, allerdings fehlen diesmal genau 2 Zahlen. **Bestimme** die fehlenden Zahlen mit Speicherplatz $\mathcal{O}(\log n)$.

Hinweis: Beschreibe jeweils dein Verfahren und erkläre **kurz**, wie und warum du damit die fehlenden Zahlen rekonstruieren kannst.

Aufgabe 4.2. (4+5)

Graph-Datenströme

Gegeben sei die Knotenmenge V eines ungerichteten Graphen G. Die Kantenmenge E wird in einer unbekannten Reihenfolge durch einen Datenstrom zur Verfügung gestellt. Sei $n := |V|$.

a) **Entwirf** einen deterministischen Algorithmus, der mit höchstens $\mathcal{O}(n \log n)$ Speicherplatz entscheidet, ob G zusammenhängend ist. (Beachte, dass eine Kante Speicherplatz $\Theta(\log n)$ benötigt.)

b) **Entwirf** einen deterministischen Algorithmus, der mit höchstens $\mathcal{O}(n \log n)$ Speicherplatz entscheidet, ob G bipartit ist.

Kommentar: Ein solches Modell wird benutzt, wenn ein Graph zu groß für den Hauptspeicher ist und die Kanten deshalb nur nach und nach aus dem Sekundärspeicher gelesen werden können.

Aufgabe 4.3. (7)

längste absteigende Teilfolgen

Erinnerung:
Für eine Folge $x_1, \ldots, x_n$ ist eine Teilfolge $x_{i_1}, \ldots, x_{i_k}$ mit $i_1 < \cdots < i_k$ *absteigend*, wenn $x_{i_1} > \cdots > x_{i_k}$ gilt. Ihre *Länge* ist die Anzahl der Elemente, hier also k. Eine *längste absteigende Teilfolge (LDS)* ist eine absteigende Teilfolge, deren Länge unter allen möglichen absteigenden Teilfolgen maximal ist.
Beispiel: Die Folge $2, 6, 4, 7, 2, 8, 3, 1$ besitzt die beiden LDSs $6, 4, 2, 1$ und $6, 4, 3, 1$ der Länge 4.

Entwirf einen Algorithmus, der die Länge eines LDS für einen Datenstrom reeller Zahlen berechnet. Die Teilfolge selbst muss dabei nicht bestimmt werden.
Sei L die Länge einer LDS. Der Speicherplatz soll durch $\mathcal{O}(L)$ beschränkt sein, wobei wir annehmen, dass jede Zahl in Platz $\mathcal{O}(1)$ gespeichert werden kann.

Aufgabe 4.4. (6 Bonuspunkte)

Sampling

Beim Reservoir-Sampling haben wir ein neues Element (t, x_t) mit Wahrscheinlichkeit T/t zur Stichprobe S hinzugefügt und ein zufälliges altes Element entfernt, wobei T die Stichprobengröße ist. Auf diese Weise erzielten wir eine Gleichverteilung der Elemente.

Wie sieht die Verteilung aus, wenn ein neues Element stattdessen mit Wahrscheinlichkeit $\gamma T/t$ für ein konstantes $\gamma > 0$ hinzugefügt wird? **Bestimme** die Wahrscheinlichkeit $\text{prob}[(k, x_k) \in S]$ zum Zeitpunkt $t > k$ approximativ für hinreichend großes k.