8. Aufgabenblatt zur Vorlesung

Höhere Algorithmik WiSe 2011/12

Wolfgang Mulzer, Claudia Dieckmann, Romain Grunert, Tillmann Miltzow, Lena Schlipf

Abgabe am 16. Dezember 2011 vor der Vorlesung in die Mitarbeiterschachteln

Aufgabe 1 Das Offline-Minimum-Problem *10 Punkte*

Beim Offline-Minimum-Problem sollen wir eine dynamische Menge $T \subseteq \{1, \ldots n\}$ verwalten, welche durch die Operationen **insert** und **extract-min** verändert wird, beginnend mit $T = \emptyset$. Wir erhalten eine Folge S von n **insert** und m **extract-min** Operationen, bei der jedes Element in $\{1, \ldots, n\}$ *genau einmal* eingefügt wird. Wir möchten die Ergebnisse aller **extract-min** Operationen finden. Das Problem ist offline, da wir die gesamte Folge S im Voraus kennen.

(a) Betrachten Sie die folgende Operationsfolge

$$4, 8, E, 3, E, 9, 2, 6, E, E, E, 1, 7, E, 5.$$

Hierbei bedeutet eine Zahl i, dass **insert**(i) ausgeführt werden soll, und E, dass **extract-min** ausgeführt werden soll. Geben Sie die Ergebnisse der einzelnen **extract-min**s an.

(b) Wir möchten nun einen Algorithmus für das Offline-Minimum Problem entwickeln. Dazu unterteilen wir die Folge S in Teilfolgen

$$I_1, E, I_2, E, I_3, E, \ldots, I_m, E, I_{m+1},$$

wobei I_j eine (möglicherweise leere) Folge von **insert**s darstellt und E eine **extract-min** Operation. Wir legen nun für jede Folge I_j eine (möglicherweise leere) Menge $K(j)$ an, welche die Zahlen aus I_j enthält. Danach verwenden wir den folgenden Algorithmus:

```
// em ist ein Feld mit m Elementen, dessen j-tes Element
// das Ergebnis des j-ten extract-mins speichern soll
for i := 1 to n do
    Finde die Menge K(j), die i enthaelt
    if j != m + 1 then
      em[j] <- i
      sei l > j der kleinste Index nach j,
        fuer den K(l) noch existiert
      vereinige K(l) mit K(j), nenne das Ergebnis
        K(l) // K(j) existiert nun nicht mehr
```

Führen Sie den Algorithmus an dem Beispiel aus (a) aus. Beweisen Sie, dass der Algorithmus korrekt ist.

(c) Beschreiben Sie, wie man den Algorithmus aus (b) effizient mit einer Union-Find Struktur implementieren kann und analysieren Sie die Laufzeit.

Aufgabe 2 Amortisierte Analyse *10 Punkte*

(a) Gegeben sei ein elektrisches Binärzählwerk mit beliebig vielen Ziffern aus der Menge $\{0, 1\}$. Das Umschalten einer Ziffer kostet eine Stromeinheit. Wie viele Stromeinheiten kostet es insgesamt, wenn man das Zählwerk von 0 bis n aufsteigend zählen lässt? Was sind die amortisierten Kosten pro Zählvorgang?

(b) Entwickeln und analysieren Sie eine Methode, um ein Array zu verwalten, welches seine Größe dynamisch ändern kann (ein solches Array ist zum Beispiel nützlich, um einen Stack zu implementieren). Die amortisierten Kosten pro Operation sollen konstant sein, und der Speicherplatz soll möglichst gut genutzt werden. Können Sie einen Trade-off zwischen der Laufzeit und der Speicherplatznutzung finden?

Aufgabe 3 Wahrscheinlichkeitsrechnung und Hashing *10 Punkte*

(a) Auf der ausgelassenen Weihnachtsfeier der Teilnehmer der HA-Vorlesung gibt es einen Julklapp (in Schwaben auch Wichteln genannt). Die Teilnehmer bringen jeweils ein Geschenk mit, und alle Geschenke werden in einen großen Sack getan. Danach zieht jeder der Partygäste zufällig ein Geschenk aus dem Sack. Was ist die erwartete Anzahl der Leute, die ihr eigenes Geschenk ziehen?

(b) Sei N die Größe einer gegebenen Hashtabelle und n beliebig. Zeigen Sie, dass für jede Schlüsselmenge K mit $|K| \geq (n-1)N + 1$ und jede Hashfunktion $h : K \to \{0, \ldots, N-1\}$ eine Menge $S \subseteq K$ mit $|S| \geq n$ existiert, so dass alle Elemente von S auf denselben Eintrag der Hashtabelle abgebildet werden. Was bedeutet das für die worst-case Laufzeit von Hashing mit Verkettung?