Technische Universität München
Fakultät für Informatik
Lehrstuhl für Effiziente Algorithmen
Prof. Dr. Ernst W. Mayr
Matthias Baumgart

Wintersemester 2007/2008
Übungsblatt 9
14. Dezember 2007

Effiziente Algorithmen und Datenstrukturen I

Abgabetermin: 21.12.2007 **vor** *der Vorlesung*

Aufgabe 1 (10 Punkte)

In der Vorlesung wurde die Laufzeit $O(k \log^* n)$ für eine Union-Find-Struktur mit *weighted union* und *Pfadkompression* bewiesen. Hierbei wurde für jeden Knoten der folgende Wert definiert:

$$\mathit{rank}(x) = \text{Höhe des Unterbaums mit Wurzel } x \text{ in der Datenstruktur } \textit{ohne} \text{ Pfadkompression.}$$

(a) Zeigen Sie, dass für alle Knoten in der Datenstruktur mit Pfadkompression gilt, dass die Werte von *rank* auf allen Pfaden von den Blättern zur Wurzel strikt steigen.

(b) Sei $\mathit{rank}_p(x)$ der Wert von *rank* des aktuellen Vaters vom Knoten x. Zeigen Sie, dass $\mathit{rank}_p(x)$ bei einem Wechsel des Vaterknotens von x nicht kleiner wird.

Aufgabe 2 (10 Punkte)

Wie in der Vorlesung eingeführt, sei eine Union-Find Datenstruktur durch eine Menge von Intrees (d.h. Bäume mit zur Wurzel gerichteten Kanten) realisiert. Es wird keine Pfadkompression verwendet. Die Union-Operation sei wie folgt definiert:

> Die Wurzel des Baums mit geringerer Höhe wird als direktes Kind an die Wurzel des anderen Baums gehängt. Sind beide Bäume gleich hoch, wird zufällig einer der Bäume ausgewählt und wie beschrieben in den anderen Baum eingehängt.

Sei size(T) bzw. height(T) die Anzahl der Knoten bzw. die Höhe eines Intrees T. Zeigen Sie, dass für alle auftretenden Intrees T in einer Folge von Operationen, die auf eine Menge von n einelementigen Mengen angewendet wird, gilt:

$$\mathrm{size}(T) \geq 2^{\mathrm{height}(T)}$$

Aufgabe 3 (10 Punkte)

Implementieren Sie eine Union-Find-Datenstruktur, die die Union- und Find-Operation in folgender Weise unterstützt:

- Beim Union zweier Intrees T_1 und T_2 wird derjenige Intree, der weniger Knoten enthält, an die Wurzel des anderen Intrees gehängt.
- Bei einem Find wird jeder Knoten auf dem Weg zur Wurzel an seinen Großvater (falls vorhanden) gehängt.

Senden Sie Ihren Quelltext an `baumgart@in.tum.de`.

Aufgabe 4 (10 Punkte)

Wir untersuchen eine alternative Implementierung von Union-Find-Strukturen: Jede der verwalteten Mengen wird als einfach verkettete Liste dargestellt, wobei zusätzlich jedes Element einen *Rückwärts-Zeiger* auf das Element am Listenanfang besitzt, welches als Repräsentant der Menge dient. Dies ermöglicht FIND-Operationen mit Kosten $O(1)$. Für UNION(S_1, S_2) wird die kürzere der beiden Listen an das Ende der längeren Liste angehängt, und die Rückwärtszeiger der Elemente der kürzeren Liste werden aktualisiert. Jede Aktualisierung eines Zeigers kostet $O(1)$ Zeit.

(a) Beginnend mit n einelementigen Mengen führen wir $n-1$ UNION-Operationen aus, bis wir bei einer einzigen n-elementigen Menge angelangt sind. Geben Sie eine Laufzeitabschätzung hierfür an.

(b) Geben Sie eine Implementierung an, die ein n-elementiges Feld statt einer linearen Liste verwendet.