Prof. Andreas Goerdt
Frank Schädlich

TU Chemnitz
Sommersemester 2003

Theoretische Informatik III

5. Übung

1. Aufgabe

Wir untersuchen, wie sich der Splaybaum auf dem Anfangsbeispiel der Vorlesung verhält:

Fügen Sie die Elemente $n, n-1, n-2, ..., 1$ in einen anfänglich leeren Splaybaum ein. Führen Sie danach `Find`(n) aus. Überlegen Sie sich, daß die Kosten der zugehörigen Folge von Operationen tatsächlich

$$O((n+1) \cdot \log n)$$

betragen. Beachten Sie, daß in diesem Fall $(n+1) = \sharp$Operationen gilt. Überlegen Sie sich, wie die darauffolgenden `Find`(n-1), ... , `Find`(1) bearbeitet werden. Wie kann danach wieder ein „ganz dünner“ Baum entstehen?

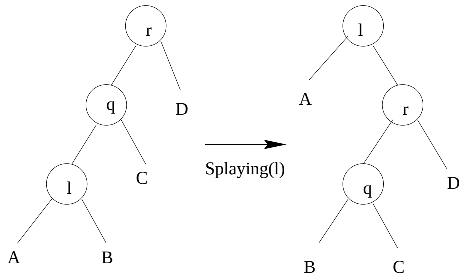
2. Aufgabe

Zeigen Sie folgenden Satz: Sei σ eine Folge von m Operationen `Find`, `Insert`, `Delete` sowie n die Maximalzahl von Elementen im Splay-Baum S bei der Anwendung von σ. Es gilt dabei Kosten(σ) =O($(m+n) \cdot \log n$).

Beachten Sie, daß $O((m+n) \cdot \log n) = O(m \cdot \log n)$ für $m \geq n$ gilt. Der Satz unterscheidet sich von dem der Vorlesung dadurch, daß der anfängliche Splay-Baum nicht leer sein muß sondern bis zu n Elementen enthalten darf. Dies gilt ebenfalls für jeden weiteren Baum, der während der Operationsfolge gebildet wird.

3. **Aufgabe**

Wir nehmen an, die Operation Splaying wäre im Zick-Zick-Fall analog zum Zick-Zack-Fall definiert. D. h.



(a) Geben Sie den Knoten r, q, l die individuellen Gewichte 1 und den Teilbäumen A, B, C, D so totale Gewichte, daß für die amortisierte Zeit a von Splaying *nicht* die im Satz der Vorlesung bewiesene Eigenschaft

$$a \leq 3(R_{S_i}(l) - R_{S_{i-1}}(l))$$

gilt.

(b) Überlegen Sie sich eine Folge von n Operationen `Find`, `Insert` und `Delete`, so daß in dem hier betrachteten Fall der Satz der Vorlsesung *nicht* gilti, die Laufzeit somit nicht $O(n \cdot \log n)$ beträgt.