Institut für Informatik
der Universität München
Prof. M. Hofmann
Dipl.-Inf. Hermann Gruber

SS 2007

26. Juni 2007

Übungen zur Vorlesung
Effiziente Algorithmen

Blatt 9

Aufgabe H-31:

a) Verwenden Sie den in der Vorlesung vorgestellten Algorithmus, um eine optimale Klammerung zur Multiplikation von 6 Matrizen $M_1, \ldots, M_6$ mit den folgenden Dimensionen anzugeben:

$M_1 : 5 \times 10,\ M_2 : 10 \times 3,\ M_3 : 3 \times 12,\ M_4 : 12 \times 5,\ M_5 : 5 \times 50,\ M_6 : 50 \times 6$

Geben Sie auch die Tabelle an, die der Algorithmus aufstellt.

b) Benutzen Sie den in der Vorlesung vorgestellten Algorithmus, um eine längste gemeinsame Teilfolge der DNA-Sequenzen `CGCGTACT` und `GTACCATGT` zu finden.

(6 Punkte)

Aufgabe H-32: Für eine Folge von Operationen auf einer Datenstruktur gelte, dass die Kosten c_i der i-ten Operation

$$c_i = \begin{cases} i & \text{falls } i = 2^k \text{ für ein } k \in \mathbb{N} \\ 1 & \text{sonst} \end{cases}$$

sind. Berechnen Sie eine einfache Formel für die *exakten* Gesamtkosten für eine Folge von n Operationen mit Hilfe der Aggregat-Methode.
(2 Punkte)

Aufgabe H-33: Erinnern Sie sich an das Beispiel des k-Bit Binärzählers aus der Vorlesung, dessen Gesamtkosten für n-faches Inkrementieren $O(n)$ sind. Erweitern Sie diesen Zähler um eine entsprechende Operation DECREMENT, die Sie in Pseudocode beschreiben. Hierbei dürfen Sie annehmen, dass per Definition DECREMENT$(0)=0$ festgelegt ist. Geben Sie mit Hilfe der Aggregatmethode möglichst gute asymptotische obere und untere Schranken an die Gesamtkosten für n Operationen im schlechtesten Fall (DECREMENT und INCREMENT gemischt).
(4 Punkte)

Hinweis: Überlegen Sie sich eine geeignete Sequenz von n Operationen, die einen *möglichst* schlechten Fall realisiert, und vergleichen Sie deren Laufzeit mit einer geeigneten oberen Schranke für den schlechtesten Fall.

Aufgabe H-34: Wir haben bisher nur Hashtabellen fester Größe betrachtet. Allerdings gibt es hierbei Probleme, wenn durch das Einfügen unvorgesehen vieler Schlüssel der Lastfaktor α auf einmal zu groß wird. Ist die Anzahl der zu speichernden Schlüssel nicht von vornherein bekannt, ist eine mögliche Strategie die folgende:

Angenommen, wir verwenden die Multiplikationsmethode für eine offen adressierte Hashtabelle der Größe m. Sobald der Lastfaktor α größer gleich $3/4$ ist, werden alle nichtleeren Einträge der Hashtabelle "eingesammelt", und es wird eine neue Tabelle mit Größe $m' = \frac{3}{2} \cdot m$ angelegt, worin die vorhandenen Schlüssel eingefügt werden. Mit dieser Hashtabelle verfährt man analog, sobald der Lastfaktor $3/4$ erreicht, usw.

a) Analysieren Sie asymptotisch die (erwarteten) Gesamtkosten dieser Strategie für das Einfügen von n Elementen in die Hashtabelle, Uniform Hashing vorausgesetzt. Die Tabelle hat anfangs die Größe $m = 16$.

b) Was ist der Nachteil obiger Strategie? Geben Sie mindestens einen Grund an, warum man in einigen Fällen anstatt obiger Strategie dennoch Rot-Schwarz-Bäume verwendet um ganzzahlige Schlüssel zu verwalten, die ja die gleichen Dienste wie Hashtabellen anbieten: Suchen, Einfügen, und Löschen.

(6 Punkte)

Abgabe bis Dienstag, 3. Juli, 09.00 Uhr im dafür vorgesehenen Briefkasten in der Theresienstraße.