Prof. Dr. Volker Linnemann
Dr. Sven Groppe
Universität zu Lübeck
Institut für Informationssysteme

Lübeck, den 15.5.2008

Vorlesung „Algorithmen und Datenstrukturen“

Sommersemester 2008

5. Übungsblatt

1. Hashing (5 Punkte)

Sei H eine endliche Menge von Hash-Funktionen mit Signatur $U \rightarrow \{ 0, 1, \ldots, m\text{-}1 \}$. H wird universell genannt, falls für jedes Paar von verschiedenen Schlüsseln $x, y \in U$ die Anzahl der Hash-Funktionen $h \in H$ mit $h(x) = h(y)$ genau $|H|/m$ ist. Mit anderen Worten, die Chance einer Kollision zwischen x und y mit $x \neq y$ ist genau $1/m$ bei einer zufällig gewählten Hash-Funktion $h \in H$, also genau so groß wie die Chance einer Kollision bei Hash-Funktionen, die alle Elemente aus U zufällig auf $\{ 0, 1, \ldots, m\text{-}1 \}$ abbilden. Bemerken Sie, dass für die letztgenannte Art von Hash-Funktionen eine Tabelle der Größe $|U|$ notwendig wäre, um die Abbildungen von U auf $\{ 0, 1, \ldots, m\text{-}1 \}$ zu verwalten.

Die folgende Klasse H_{prim} von Hash-Funktionen ist universell:

Sei m eine Primzahl. Wir zerlegen einen Schlüssel x in $r + 1$ Unterkomponenten, so dass $x = < x_0, x_1, \ldots, x_r >$. Die einzige Bedingung ist, dass der maximale Wert einer Unterkomponente kleiner als m sein muss. Sei nun weiterhin $a = < a_0, a_1, \ldots, a_r >$ eine Sequenz von $r + 1$ Elementen, die zufällig aus $\{ 0, 1, \ldots, m\text{-}1 \}$ gewählt sind. Dann sei eine entsprechende Hash-Funktion $h_a \in H_{prim}$ durch $h_a(x) = (\Sigma_{i=0}^{r} a_i * x_i) \bmod m$ definiert. Bemerken Sie, dass wir durch Zahlentheorie beweisen können, dass H_{prim} universell ist.

a) Berechnen Sie die Hash-Werte von 3, 20 und 82 bezüglich der Hash-Funktion $h_{<5, 8, 9>}(x) = (\Sigma_{i=0}^{r} a_i * x_i) \bmod 13$. Eine Unterkomponente x_i habe dabei den Wertebereich $\{0, \ldots, 7\}$ und werde durch $x_i = (x \text{ div } 8^i) \bmod 8$ berechnet. (3 Punkte)
b) Zeigen Sie: Falls wir $a_i > 0$ fordern würden für jedes $a_i \in \{ a_0, a_1, \ldots, a_r \}$, dann ist H_{prim} nicht mehr universell. (2 Punkte)

2. Anwenden von Sortierverfahren (15 Punkte)

Geben Sie jeden Zwischenschritt an für das Sortieren der Folge < 89, 5, 2, 56, 42 >, wenn die folgenden Sortierverfahren angewendet werden:

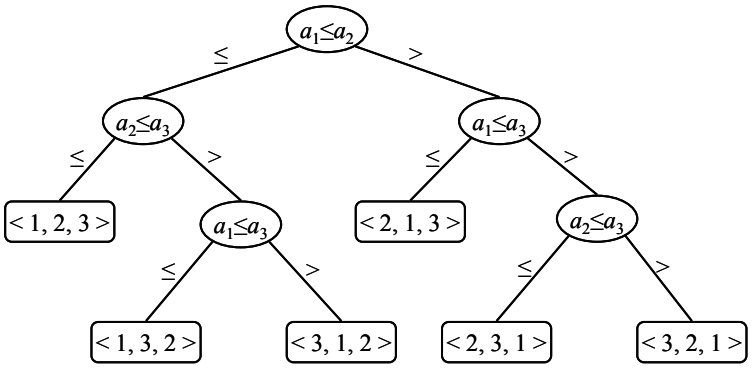
a) Direktes Aussuchen (5 Punkte)
b) Direktes Einfügen (5 Punkte)
c) Bubble Sort (5 Punkte)

3. Untere Schranke für Sortieren durch Vergleichsoperationen (5 Punkte)

Sei n die Anzahl der zu sortierenden Elemente $a_1, \ldots, a_n$. In dieser Aufgabe wollen wir die untere Schranke $\Omega(n*\log(n))$ für den schlechtesten Fall für die Klasse V von Sortierverfahren beweisen, die nur über Vergleiche zwischen zwei Elementen Informationen der relativen Ordnung innerhalb der Eingabesequenz verwenden. Zu dieser Klasse von Sortieralgorithmen gehören zum Beispiel Bubblesort, Heapsort, Quicksort, Shellsort, Mergesort, Direktes Aussuchen und Direktes Einfügen, aber nicht Bucketsort.

Ohne Einschränkung der Allgemeinheit seien alle zu sortierenden Elemente verschieden und alle Vergleiche haben die Form $a_i \leq a_j$, wenn a_i und a_j zu sortierende Elemente sind.

Die notwendigen Vergleiche für beliebige zu sortierende Elemente $a_1, \ldots, a_n$ von Sortierverfahren aus V können abstrakt durch *Entscheidungsbäume* repräsentiert werden. Zur Ermittlung einer unteren Schranke für Sortierverfahren aus V werden dabei andere Aspekte wie die zur Programmkontrolle und zur Datenbewegung ignoriert. Die unten stehende Abbildung zeigt zum Beispiel einen Entscheidungsbaum für das Sortierverfahren Direktes Einfügen für 3 zu sortierende Elemente.



Jeder innere Knoten von Entscheidungsbäumen repräsentiert dabei einen Vergleich $a_i \leq a_j$ zwischen den zu sortierenden Elementen. Jeder Blattknoten von Entscheidungsbäumen repräsentiert eine Permutation $< \pi(1), \ldots, \pi(n) >$ der zu sortierenden Elemente $a_1, \ldots, a_n$. Die Ausführung eines Sortieralgorithmus entspricht dabei einem Pfad von der Wurzel bis zu einem Blattknoten des Entscheidungsbaumes. Bei jedem inneren Knoten wird ein Vergleich $a_i \leq a_j$ vollzogen. Der linke Teilbaum des aktuellen inneren Knotens enthält die nachfolgenden Vergleiche, falls $a_i \leq a_j$ gilt, und der rechte Teilbaum enthält die nachfolgenden Vergleiche, falls $a_i > a_j$ gilt. Bei Erreichen eines Blattknotens hat das Sortierverfahren die Ordnung $a_{\pi(1)} \leq a_{\pi(2)} \leq \ldots \leq a_{\pi(n)}$ bestimmt.

Beweisen Sie, dass jeder Entscheidungsbaum, der n Elemente sortiert, die Höhe $\Omega(n*\log(n))$ hat und damit jedes Sortierverfahren aus der Klasse V die Worst-Case-Laufzeit $\Omega(n*\log(n))$ besitzt.

Bemerkung: Es gilt $n! > (n/e)^n$ (Approximation von Stirling), wobei e die Eulersche Zahl 2,71828… repräsentiert.

Bemerkungen:

- Jede Seite soll oben rechts den Namen der Abgebenden und die Übungsgruppennummer (wichtig!) enthalten.
- Lösungen für die Übungsaufgaben sind (in der Regel) zu zweit abzugeben.
- Kommentieren Sie Ihre Lösungen! Besteht eine Lösung aus mehreren Zetteln, so sind diese zusammen zu heften. Bitte keine Hüllen, Mappen, o.ä..
- Bitte schicken Sie *Programmieraufgaben zusätzlich zur Abgabe auf Papier in elektronischer Form per Email* an Ihren jeweiligen Tutor.
- Kommentieren Sie ihren Quelltext bei Programmieraufgaben. Dabei sollen keine Trivialitäten kommentiert werden, also bitte keine Kommentare wie

  ```
  x=5; // Wir weisen nun der Variablen x den Wert 5 zu
  ```

  sondern sinnvolle Kommentare, die Ideen des Quelltextabschnittes beschreiben oder auf Unteraufgaben (z. B. a), b), …) hinweisen.
- **Hinreichende Bedingung für die Zulassung zur Klausur:** 50% der erreichbaren Punkte bei jedem Übungszettel (bis auf zwei) und einmaliges Vorrechnen in der Übung
- **Zertifikatskriterium:** Das Bestehen der Klausur am Ende des Semesters

Abgabetermin: Donnerstag, 22.5.2008, nach der Vorlesung