Vorlesung Grundlegende Algorithmen
Wintersemester 2002/2003
(3-stdg. Vorl. mit 2-stdg. Übung)

Lehrstuhl für Effiziente Algorithmen
Technische Universität München
Dr. Sven Kosub
Dipl.-Ing. Dipl.-Math. Thomas Bayer

Inhalt

1. Einführung

1.1 Beispiel: Fibonacci-Zahlen
(rekursive Definition der Fibonacci-Zahlen)

1.1.1 Rekursive Berechnung
(rekursiver Algorithmus; Definition Laufzeit als Anzahl arithmetischer Operationen; exponentielle Laufzeit)

1.1.2 Iterative Berechnung
(iterativer Algorithmus; lineare Laufzeit)

1.2 Grundlegende Konzepte

1.2.1 Registermaschinen (RAM)
(Definition: RAM durch Transport-, arithmetische und Sprungbefehle; Beispiel Multiplikation)

1.2.2 Komplexitätsmaße
(uniforme und logarithmische Eingabegröße; uniforme und logarithmische Zeitkomplexität; logarithmische RAM-Kosten; uniforme und logarithmische Zeitbeschränktheit; RAM-Beispiel Multiplikation)

1.2.3 Asymptotisches Verhalten von Funktionen
(Definition von $O, \Omega, \Theta, o, \omega$ mittels Quantoren (nicht durch Grenzwerte); Schreibweisen; Definition: konstant, logarithmisch, polylogarithmisch, linear, quadratisch, polynomiell, exponentiell)

2. Sortieren
(Problembeschreibung; Vereinfachung: Sortieren nur Tupel natürlicher (ganzer) Zahlen)

2.1 Elementare Sortierverfahren

2.1.1 Sortieren durch Auswahl
(Idee und Algorithmus SELECTIONSORT; Komplexität: genau $\binom{n}{2}$ bzw. $\Theta(n^2)$ Vergleiche)

2.1.2 Sortieren durch Einfügen
(Idee und Algorithmus INSERTIONSORT; Komplexität: max. $\binom{n}{2}$ bzw. $O(n^2)$ Vergleiche)

2.2 Sortieren durch Mischen
(Mischen zweier sortierter Tupel mit max. $n + m - 1$ Vergleichen)

2.2.1 Rekursives Sortieren durch Mischen

(Idee und rekursiver Algorithmus MERGESORT; Komplexität: max. $n\lceil\log n\rceil$ bzw. $O(n\log n)$ Vergleiche)

2.2.2 Iteratives Sortieren durch Mischen

(Idee und iterativer Algorithmus MERGESORT; Komplexität: max. $n\lceil\log n\rceil$ bzw. $O(n\log n)$ Vergleiche)

2.3 HEAPSORT

(Definition: (binärer) Baum, Level, Tiefe, fast vollständiger Baum, Heap; Algorithmus HEAPIFY; Korrektheit von MAKEHEAP (als Übung); Idee und Algorithmus HEAPSORT; Komplexität: max. $2n\log n + \frac{13}{2}n$ bzw. $O(n\log n)$ Vergleiche)

2.4 QUICKSORT

(Definition: Rang, Median; Pivot-Element und Aufteilungsschritt; Algorithmus; Komplexität: max. $\binom{n}{2}$ bzw. $O(n^2)$ Vergleiche im schlechtesten Fall, max. $n\log n + O(n)$ Vergleiche im besten Fall)

2.5 Untere Schranke für vergleichsbasiertes Sortieren

(Definition: Entscheidungsbaum; $n!$ Blätter in Entscheidungsbaum; Mindesthöhe $\lceil\log n\rceil$ von binärem Baum mit n Knoten; Komplexität von Sortieralgorithmen: $\frac{1}{2}n\log n$ bzw. $\Omega(n\log n)$ Vergleiche im schlechtesten Fall)

3. Selektieren

(Problembeschreibung; Vereinfachung: paarweise verschiedene Zahlen)

3.1 QUICKSELECT

(Idee und Algorithmus; Komplexität: max. $\binom{n}{2}$ bzw. $O(n^2)$ Vergleiche im schlechtesten Fall, max. $4n$ bzw. $O(n)$ Vergleiche im Mittel)

3.2 BFPRT-Algorithmus

(Idee (Median der Mediane von fünf Elementen) und Algorithmus; Komplexität: max. $26n$ Vergleiche bzw. $O(n)$-Vergleiche; Komplexität von QUICKSORT mit BFPRT: $O(n\log n)$ Vergleiche)

4. Suchen

(Problemstellungen; Operationen: IsMember, Insert, Delete)

4.1 Ausnutzung von Sortierung

4.1.1 Lineare Suche

(Ideen und Algorithmen; Komplexität für alle Operationen: $O(n)$)

4.1.2 Binäre Suche

(Ideen und Algorithmen; Komplexitäten: $O(\log n)$ für IsMember, $O(n)$ für Insert, Delete)

4.2 Hashing

4.2.1 Hashfunktionen

(Definition: Hashfunktion, Kollision, ideale Hashfunktion; Wahrscheinlichkeit für Auftreten von Kollisionen: $1 - o(1)$ für mindestens $\omega(\sqrt{n})$ Einträge; Beispiele für Hashfunktionen: Teiler-, Multiplikations- und Additionsmethode)

4.2.2 Strategien zur Kollisionsauflösung

(Definition: offene bzw. geschlossene Auflösung, Sondierung, Füllfaktor α; Verkettung (CHAINING) mit Anzahl Sondierungen im Mittel: $1+\frac{\alpha}{2}$ für erfolgreiches IsMember, $1+\alpha$ für erfolgloses IsMember; Lineares Sondieren (LINEAR PROBING) mit Anzahl Sondierungen im Mittel (ohne Beweis): $\frac{1}{2}\left(1+\frac{1}{1-\alpha}\right)$ für erfolgreiches $\frac{1}{2}\left(1+\frac{1}{(1-\alpha)^2}\right)$ für erfolgloses IsMember; Quadratisches Sondieren (QUADRATIC PROBING) mit Anzahl Sondierungen im Mittel (ohne Beweis): $1-\frac{\alpha}{2}-\ln(1-\alpha)$ für erfolgreiches bzw. $\frac{1}{1-\alpha}-\alpha-\ln(1-\alpha)$ für erfolgloses IsMember; Doppeltes Hashing (DOUBLE HASHING) mit Anzahl Sondierung im Mittel (ohne Beweis): $\frac{1}{\alpha}-\ln(1-\alpha)$ für erfolgreiches IsMember und $\frac{1}{1-\alpha}$ für erfolgloses IsMember)

4.2.3 Universelles Hashing

(Definition: universelle Familie von Hashfunkionen; im Erwartungswert max. eine Kollision von zwei Elementen; Beispielfamilie und Nachweis der Universalität)

4.3 Binäre Suchbäume

(Definition: binärer Suchbaum; Idee und Algorithmen; Komplexitäten für alle Operationen: $O(h(T))$)

4.4 AVL-Bäume

(Definition: höhenbalanciert, AVL-Baum; AVL-Baum der Höhe h hat mindestens $\left(\frac{1+\sqrt{5}}{2}\right)^h$ Knoten; Komplexität: $O(\log n)$ für IsMember; einfache und doppelte Rotation für Insert; Komplexität: $O(\log n)$ für Insert; einfache und doppelte Rotation für Delete; Komplexität: $O(\log n)$ für Delete)

4.5 (a, b)-Bäume

(Definition: allgemeine Suchbäume, (a, b)-Baum; Höhe von (a, b)-Baum mit n Blättern min. $\lceil \log_b n \rceil$ und max. $\lceil \log_a \frac{n}{2} \rceil + 1$; Ideen und Algorithmen für IsMember, Insert, Delete; Komplexitäten für alle Operationen: $O(\log n)$)

5. Texte

(Definitionen: Alphabet, Wort, Teilwort, Präfix, Suffix, Rand)

5.1 Der KNUTH-MORRIS-PRATT-Algorithmus

(Definition: Mismatch, zulässiger Shift; naiver Algorithmus; Idee und Algorithmus; Komplexitäten: $2n-m+1$ Vergleiche für KMP ohne Ränderberechnung, $2m-1$ Vergleiche für Ränderberechnung, $2n+m$ Vergleiche insgesamt)

5.2 Der BOYER-MOORE-Algorithmus

(Idee und Algorithmus mit Good-Suffix-Rule; Komplexität: $3(n+m)$ Vergleiche; Einbindung der Bad-Character-Rule)

5.3 Datenkompression

(Existenz nicht komprimierbarer Texte)

5.3.1 Die HUFFMAN-Kodierung

(Definition: Kodierung, Code, Codewort, Codewortlänge, mittlere Codewortlänge, Präfix-Code, optimaler Präfix-Code; Idee und Algorithmus; Beweis der Optimalität)

5.3.2 Der LEMPEL-ZIV-WELCH-Algorithmus

(LZ77, LZ78; Idee und Algorithmus für LZW)

Literatur

[CLRS01] THOMAS H. CORMEN, CHARLES E. LEISERSON, RONALD L. RIVEST und CLIFFORD STEIN. *Introduction to Algorithms.* 2. Auflage, MIT Press, Cambirdge, MA, 2001.

[Heu00] VOLKER HEUN. *Grundlegende Algorithmen: Einführung in den Entwurf und die Analyse effizienter Algorithmen.* Vieweg-Verlag, Braunschweig-Wiesbaden, 2000.

[Meh88] KURT MEHLHORN. *Datenstrukturen und effiziente Algorithmen,* Band 1 *Sortieren und Suchen.* Leitfäden und Monographien der Informatik. 2. Auflage, B. G. Teubner, Stuttgart, 1988.

[Sch01] UWE SCHÖNING. *Algorithmik.* Spektrum Akademischer Verlag, Heidelberg, 2001.

[Sed02] ROBERT SEDGEWICK. *Algorithmen.* Pearson Education, München, 2002.

[Wag94] KLAUS W. WAGNER. *Einführung in die Theoretische Informatik: Grundlagen und Modelle.* Springer-Verlag, Berlin-Heidelberg-New York, 1994.

Literaturhinweise

zu Abschnitt 1.1: [Heu00, S. 1–5].
zu Abschnitt 1.2.1: [Wag94, S. 21–23].
zu Abschnitt 1.2.2 und 1.2.3: [Heu00, S. 13–22]; die logarithmischen Kostenmaße sind ähnlich in [Meh88, S. 5–6] definiert.
zu Abschnitt 2 (vor 2.1): [Heu00, S. 25–26].
zu Abschnitt 2.1: [Heu00, S. 26–31].
zu Abschnitt 2.2: [Heu00, S. 32–39].
zu Abschnitt 2.3: [Heu00, S. 39–51].
zu Abschnitt 2.4: [Heu00, S. 54–58].
zu Abschnitt 2.5: [Heu00, S. 68–72].
zu Abschnitt 3: [Heu00, S. 83–90].
zu Abschnitt 4: [Heu00, S. 113–143]; Beweise für die erwartete Anzahl von Sondierungen beim Hashing (Abschnitt 4.2.2) sind in [CLRS01, S. 241–244] zu finden.
zu Abschnitt 5 (vor 5.1), 5.1 und 5.2: [Heu00, S. 205–214]; ein Beweis für die Anzahl der Vergleiche des BOYER-MOORE-Algorithmus ist in [Heu00, S. 214–218] zu finden.
zu Abschnitt 5.3 (vor 5.3.1) und 5.3.1: [Heu00, S. 227–232]; Beispiele und Implementierungshinweise in [Sed02, S. 371–383].
zu Abschnitt 5.3.2: [Sch01, S. 260–265].