Algorithmische Mathematik I

Wintersemester 2011 / 2012
Prof. Dr. Sven Beuchler
Peter Zaspel

universitätbonn

Übungsblatt 3.

Abgabe am **09.11.2011**.

Aufgabe 1. (Tschebyscheff–Polynome)

a) Geben Sie die ersten sechs Tschebyscheff–Polynome $T_0, T_1, \ldots, T_5$ mithilfe der zugehörigen Drei–Term–Rekursion an.

b) Beweisen Sie, dass der höchste Koeffizient des n-ten Tschebyscheff–Orthogonalpolynoms $2^{n-1}x^n$ ist $(n \geq 1)$.

(5 Punkte)

Aufgabe 2. (Gesetze der Rückwärtsanalyse)

Bei der Rückwärtsanalyse wird das Resultat einer Rechnung als exaktes Ergebnis für gestörte Operanden interpretiert, also $a \dot{*} b = (a * b)(1 + \xi)$ für $* \in \{+, -, \cdot, /\}$. Welche der folgenden Gesetze gelten, wenn man ξ für alle Operanden als konstant annimmt?

- Kommutativgesetze: $a + b = b + a$ bzw. $a \cdot b = b \cdot a$
- Assoziativgesetze: $(a + b) + c = a + (b + c)$ bzw. $(a \cdot b) \cdot c = a \cdot (b \cdot c)$
- Distributivgesetze: $a \cdot (b + c) = a \cdot b + a \cdot c$ bzw. $(a + b) \cdot c = a \cdot c + b \cdot c$

(5 Punkte)

Aufgabe 3. (Polynomauswertung)

Wir wollen nun eine Fehleranalyse bei der Auswertung von Polynomen $y = c_0 + c_1 x + \ldots + c_n x^n$ durchführen. Bei der Rückwärtsanalyse ist das gestörte Resultat der Auswertung $\tilde{y}$ das Ergebnis einer exakten Rechnung für gestörte Koeffizienten $\tilde{c}_i$, also $\tilde{y} = \tilde{c}_0 + \tilde{c}_1 x + \ldots + \tilde{c}_n x^n$. Im Folgenden sei der Fehler für alle Operationen konstant gleich ξ.

a) Geben Sie die Koeffizienten $\tilde{c}_i$ an, für den Fall, dass das Polynom beginnend mit der niedrigsten Potenz ausgewertet wird. Linearisieren Sie dabei rechtzeitig, also zum Beispiel ist $(1 + \xi)(1 + \xi) \approx 1 + 2\xi$.

b) Beginnen Sie nun mit der höchsten Potenz. Welche ist die bessere Strategie?

(5 Punkte)

Aufgabe 4. (Sortierverfahren)

a) Wenden sie den Sortieralgorithmus *Bubble-Sort* auf die Zahlensequenz $5, 2, 3, 4, 1$ an. Und schreiben sie alle Vertauschungsschritte auf.

b) Geben sie die Zahl der notwendigen Vertauschungen und Vergleiche für das Bubble-Sort-Verfahren an.

c) Wenden sie nun den Sortieralgorithmus *Merge-Sort* auf die Zahlensequenz $5, 2, 3, 4, 1$ an. Und schreiben sie alle Schritte auf.

d) Geben sie für Merge-Sort die Zahl der notwendigen Vergleiche (zwischen Zahlen der zu sortierenden Liste) sowie die Rekursionstiefe (ohne den ersten Aufruf) an.

(5 Punkte)

Programmieraufgabe 1. (Verkettete Liste)

Bisher haben wir im wesentlichen als Datenstrukturen nur Arrays betrachtet. Der große Vorteil bei Arrays liegt darin, dass sie bei einer unveränderlichen Anzahl an Elementen am effizientesten sind, was den Speicherverbrauch betrifft. Ebenso kann man auf jedes beliebige Element eines Arrays in konstanter Zeit zugreifen. Allerdings ergeben sich Schwierigkeiten, wenn man eine dynamische Anzahl an Elementen in einer Liste verwalten möchte.

a) Welche Schwierigkeiten sind das?

b) Falls das Konzept von verketteten Listen noch nicht bekannt ist, informieren Sie sich hierüber bitte über geeignete Literatur.

c) Implementieren sie einzelne Listenelemente als Struct mit dem Namen `element`. Der gespeicherte Wert soll ein `int`-Wert sein.

d) Schreiben sie eine Funktion `add(element** head, int entry)`, die (nach Übergabe eines Doppel-Pointers `head` auf den Listenkopf), ein Element mit dem Eintrag `entry` an das Ende der Liste einfügt. (Beachten Sie die korrekte Behandlung des Falles, dass gilt `*head == 0`, also dass die Liste kein Element enthält.)

e) Schreiben sie eine Funktion, die unter Angabe des Eintrags `entry` ein Element aus der Liste löscht. Auch hier soll der Fall der leeren Liste geeignet behandelt werden.

f) Implementieren Sie das Bubble-Sort-Verfahren und verwenden Sie dabei die verkettete Liste als Datenstruktur. Es kann hier zum Beispiel sinnvoll sein sich zusätzlich eine Methode zum Vertauschen einzelner Einträge zu implementieren. (Sie können davon ausgehen, dass alle zu sortierenden Elemente paarweise verschieden sind.)

g) Testen Sie Ihre Implementierung anhand des Beispiels aus Aufgabe 4.

h) Machen Sie sich mit Hilfe des Internets mit dem Tool `valgrind` vertraut und überprüfen Sie mit diesem, ob Ihr Code fehlerfrei den Speicher verwendet.

i) Wie effizient ist es, eine verkettete Liste für den Bubble-Sort-Algorithmus zu verwenden? Warum?

(15 Punkte)

Die Abgabe der Programmieraufgaben erfolgt in den CIP-Pools in der Woche vom 14.11. bis 18.11.2011. Die Listen für die Anmeldung zu den Abgabe- Terminen hängt in der Woche vom 07.11. bis 11.11.2011 aus.