Vorwort

Die Informatik als Wissenschaft der Informationsverarbeitung gewinnt in fast allen Lebensbereichen mehr und mehr an Bedeutung. Die weltweite Vernetzung durch das Internet hat diese Entwicklung noch einmal entscheidend beschleunigt. So sind die Errungenschaften der Informatik heute allgegenwärtig in Produktion, Logistik, Verkehr, Medizin und in den Medien. Große wissenschaftliche Erfolge wie etwa die Sequenzierung des menschlichen Genoms wären ohne die Informatik nicht möglich gewesen. Auch im privaten Bereich scheint die Informatik omnipräsent zu sein. Ihre Errungenschaften stecken nicht nur im häuslichen Computer. So gut wie alle modernen Haushaltsgeräte – von der Waschmaschine über den Toaster bis zum Fernseher – werden durch eingebettete Systeme gesteuert. In fast jedem heute vom Band laufenden PKW finden sich gleich mehrere eingebettete Systeme, die nicht nur Motor und Bremssystem, sondern auch Klimaanlage und Fensterheber steuern – in Zukunft wohl auch das automatische Einparksystem. Selbst bei der zwischenmenschlichen Kommunikation vertrauen wir uns mehr und mehr der Informatik an, indem wir moderne Formen des Informationsaustausches wie E-Mails und SMS nutzen.

Die rasante Entwicklung der Informatik, die wir in den letzten Jahrzehnten beobachten konnten und die sich wahrscheinlich auch in der nächsten Zeit unvermindert fortsetzen wird, ist nicht nur auf die stetig fortschreitende Verbesserung der Hardware, sondern insbesondere auch auf die permanente Neu- und Weiterentwicklung der Software in Form immer effizienterer Programme zurückzuführen. Die größten Verbesserungen beruhen dabei auf neu entwickelten Algorithmen, hinter denen häufig clevere und kreative Ideen stecken. Algorithmen sind geschickte Verfahren, die vorgegebene Probleme auf unterschiedlichste Weise lösen. Dabei handelt es sich nicht nur um mathematische Aufgaben, die sich beispielsweise auf das Rechnen mit Zahlen beziehen, sondern auch um andere, ganz alltägliche Problemstellungen, bei denen räumli-

che Orientierung, logischer Spürsinn oder geschicktes Verhandeln gefragt sind, beispielsweise um Fragestellungen der folgenden Art:

- Wie lässt sich der kürzeste Weg zwischen zwei Orten ermitteln?
- Wie sollten Seeräuber eine Schatzkarte aufteilen, bzw. Bankangestellte den Geheimcode des Tresors?
- Wie können mehrere hungrige Partygäste einen Kuchen gerecht untereinander aufteilen?

Dieses Buch unternimmt einen umfangreichen Streifzug durch die faszinierende Welt der Algorithmen. Es verlangt keine besonderen Vorkenntnisse, so dass Schülerinnen und Schüler ab der Mittelstufe und auch Informatikinteressierte Laien neue und überraschende Einblicke gewinnen können. In 43 Artikeln von Informatikern, die an Universitäten im In- und Ausland lehren, werden wichtige und besonders elegante Algorithmen anschaulich und verständlich erklärt.

Die Initiative zur Erstellung einer derartigen Algorithmensammlung geht zurück auf Professor Dr. Volker Claus aus dem Vorstand des *Fakultätentags Informatik*, der Vereinigung der Informatikfakultäten an deutschen Universitäten. Seine Idee war es, im Informatikjahr 2006 wöchentlich einen Algorithmus herauszugreifen und diesen im Internet zu präsentieren mit dem Ziel, Schülerinnen und Schülern zu vermitteln, welche kreativen Ideen und Konzepte hinter Algorithmen stecken und welche Faszination von der Informatik ausgehen kann. Diese Idee wurde in der Initiative *Algorithmus der Woche* umgesetzt, in der Woche für Woche von März bis Dezember 2006 jeweils ein Artikel ins Netz gestellt wurde. Für dieses Buch wurde jeder dieser Artikel gründlich und umfassend überarbeitet. Die Autoren haben zahlreiche Verbesserungsvorschläge der Herausgeber aufgenommen und auch viele weitere Ergänzungen eingebracht. Insbesondere wurden Querbezüge zu anderen Artikeln der Sammlung eingearbeitet, und am Ende jedes Artikels gibt es Hinweise auf weiterführende Literatur.

An der umfangreichen redaktionellen Arbeit zur Erstellung dieses Buches haben insbesondere Marcel Ochel und Heiko Röglin mitgewirkt. An der redaktionellen Arbeit zum *Algorithmus der Woche*, die die Grundlage für dieses Buch gelegt hat, waren neben diesen beiden auch Dirk Bongartz, Torsten Sattler, Katrin Twickler und Melanie Winkler beteiligt. Wir möchten allen Autoren und dem Redaktionsteam für ihr Mitwirken danken. Durch das große Engagement aller Beteiligten ist unserer Meinung nach eine ausgesprochen facettenreiche Sammlung von interessanten und informativen Artikeln entstanden, die die besondere Faszination der Informatik spürbar werden lässt.

Die Herausgeber im März 2008

2

Sortieren durch Einfügen

Wolfgang P. Kowalk

Universität Oldenburg

Schon wieder Aufräumen, dabei habe ich doch erst neulich ... Nun ja, wenn alles durcheinander liegt, dann lohnt es sich vielleicht doch, mal eben Ordnung zu schaffen; man findet alles schneller wieder. Lass uns also mal die Bücher auf dem Regal alphabetisch nach ihren Titeln sortieren, so dass man jedes gleich zur Hand hat, wenn man es braucht.

Doch wie schafft man am schnellsten Ordnung? Man kann verschiedene Ansätze verfolgen. So kann man wiederholt alle Bücher durchgehen, und jedesmal wenn man zwei aufeinanderfolgende Bücher sieht, die falsch herum stehen, vertauscht man sie. Das funktioniert, weil irgendwann keine zwei Bücher mehr verkehrt stehen, aber es kann lange dauern. Man kann auch zunächst das Buch mit dem (alphabetisch) ersten Titel heraussuchen und dieses nach ganz links ins Regal stellen. Danach sucht man sich das Buch mit dem ersten Titel aus den restlichen Büchern heraus und stellt es neben das erste Buch. Das macht man dann so weiter, bis alle Bücher sortiert sind. Auch diese Methode funktioniert, kann aber auch lange dauern, da sie sehr viel vorhandene Information immer wieder „vergisst“ und nicht ausnutzt; jedes Mal müssen immer wieder die gleichen Buchtitel gelesen werden. Versuchen wir also etwas anderes.

Die folgende Idee scheint sehr natürlich zu sein. Wir bringen zuerst die beiden Bücher ganz links im Regal in ihre richtige Reihenfolge, dann die ersten drei, dann die ersten vier, und so weiter, bis wir schließlich alle Bücher in sortierter Reihenfolge stehen haben. Dabei können wir in jeder Runde auf das vorher Erreichte zurückgreifen. Wie geht das ganz genau vor sich? Das erste Buch steht für sich allein gesehen am richtigen Platz. Jetzt nehmen wir das zweite Buch hinzu und vertauschen es mit dem ersten, falls es links von diesem stehen muss. Dann nehmen wir das dritte Buch und ordnen es in der richtigen Stelle bezüglich der ersten beiden Bücher ein. Danach nehmen wir das vierte Buch hinzu, sortieren es an die richtige Stelle unter den ersten dreien und so weiter. Allgemein nehmen wir an, dass alle Bücher links vom aktuellen Buch (Position i von links) bereits sortiert sind. Dann nehmen wir das aktuelle Buch an dieser Position, suchen seinen korrekten Platz in den Büchern links

davon und fügen es an dieser Position ein. Dazu müssen die Bücher rechts von diesem richtigen Platz etwas nach rechts verschoben werden. Nun geht es weiter mit der nächsten Position. Nachdem das letzte Buch von ganz rechts im Regal an die richtige Stelle gewandert ist, sind alle Bücher sortiert. Dieses Verfahren führt recht schnell zu einer sortierten Bücherreihe, besonders wenn wir das Verfahren „Binäre Suche“ aus Kap. 1 benutzen, um schnell den richtigen Platz des aktuellen Buches in der schon sortierten Bücherreihe links davon zu finden. Nur das Verschieben der Bücher rechts vom „richtigen Platz“ benötigt eventuell ziemlich viel Kraft. Bei einer sehr langen Buchreihe reicht irgendwann die Kraft nicht mehr aus, alle Bücher gleichzeitig zu verschieben, und man verschiebt wohl dann doch besser Buch für Buch.

Wie kann man unser intuitives Verfahren nun so umsetzen, dass es für jede Anzahl von Büchern durchführbar ist? Statt der Buchtitel schreiben wir im folgenden Nummern, weil die Verfahren dann einfacher zu erklären sind.

In Abb. 2.1 sind die fünf Bücher 1, 6, 7, 9, 11 ganz links bereits sortiert; wird das Buch mit der Nummer 5 hinzugenommen, so steht es offensichtlich falsch. Um es an die richtige Position zu bringen, können wir es zunächst mit dem Buch mit der Nummer 11 vertauschen, dann mit dem mit der Nummer 9 usw., bis es rechts vom Buch mit der Nummer 1 an seiner richtigen Position angekommen ist. Dann fahren wir mit dem Buch mit der Nummer 3 fort und sortieren es durch paarweise Vertauschungen richtig ein, usw. Offenbar kommen dadurch alle Bücher nacheinander an ihren richtigen Platz (siehe Abb. 2.2).

Wie kann man so etwas jetzt programmieren? Das folgende Programm macht das! Es benutzt ein Zahlenfeld (oder Array) A, dessen Zellen durchnummeriert sind. Unter $A[i]$ versteht man den Wert an der i-ten Stelle des Feldes A. Wenn wir n Bücher haben, so brauchen wir ein Zahlenfeld der Länge n mit den Stellen $A[1]$, $A[2]$, $A[3]$, …, $A[n-1]$, $A[n]$, um alle Buchtitel bzw. Buchnummern zu speichern. Der Algorithmus sieht dann folgendermaßen aus.

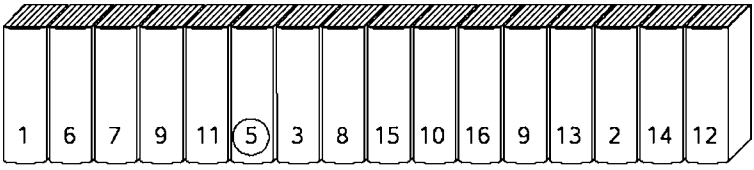

Abb. 2.1. Die ersten fünf Bücher sind sortiert

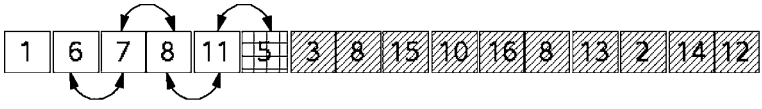

Abb. 2.2. Buch „5“ wird an den richtigen Platz gebracht

Benachbarte Bücher vertauschen:

```
1   Gegeben: A: Feld mit n Einträgen
2   for i := 2 to n do
3       j := i;      // das Buch an Position i ist aktuell
          // solange korrekte Position des aktuellen Buchs noch nicht erreicht
4       while j ≥ 2 and A[j − 1] > A[j] do
5           Hand := A[j];      // tausche aktuelles Buch mit linkem Nachb.
6           A[j] := A[j − 1];
7           A[j − 1] := Hand;
8           j := j − 1
9       endwhile
10  endfor
```

Wie lange dauert jetzt wohl das Aufräumen? Nehmen wir einmal den schlechtesten Fall an. Dann sind alle Bücher genau verkehrt herum aufgestellt, also das mit der kleinsten Nummer steht ganz rechts, daneben das mit der zweitkleinsten Nummer usw. Was macht unser Algorithmus dann? Er fängt von links an und vertauscht das zweite Buch mit dem ersten, das dritte mit den ersten beiden, das vierte mit den ersten dreien, usw., bis schließlich das letzte mit $n-1$ Büchern zu vertauschen ist. Die Anzahl der Vertauschungen ist also

$$1 + 2 + 3 + \ldots + (n-1) = \frac{n \cdot (n-1)}{2} \; .$$

Diese Formel lässt sich leicht anhand von Abb. 2.3 einsehen. Es gibt in dem Rechteck $n \cdot (n-1)$ Felder, und davon wird gerade die Hälfte für das Vergleichen

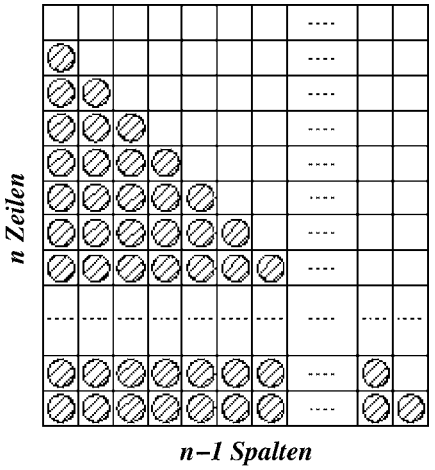

Abb. 2.3. Berechnung der Anzahl der Vertauschungen

und Vertauschen verwendet. Das Bild stellt aber den absolut schlechtesten Fall dar. Für den allgemeinen Fall überlegen wir: Um das Buch an Stelle i einzusortieren, braucht man keinesfalls mehr als $i-1$ Vertauschungen. Also gibt es für keine Ausgangssituation mehr als $\frac{1}{2}n(n-1)$ Vertauschungen. Unser Algorithmus ist umso besser, je mehr Bücher bereits richtig stehen, was z. B. der Fall ist, wenn die Bücherreihe schon einmal sortiert worden ist und einige herausgenommene Bücher einfach wahllos wieder zurückgestellt worden sind, wie das vielleicht dein kleiner Bruder gerne tut!

Sicherlich hast du schon bemerkt, dass unser Sortierverfahren umständlicher ist, als es eigentlich sein muss. (Genau genommen ist es eine spezielle Variante des am Anfang erwähnten Verfahrens, immer wieder benachbarte Bücher zu vertauschen, wenn sie falsch herum stehen ...) Wie wir schon oben überlegt haben, muss man ja nicht unbedingt die Bücher durchtauschen, sondern man kann auch einfach die schon einsortierten Bücher nach rechts verschieben, bis der Platz an der richtigen Position des aktuell einzusortierenden Buches geschaffen worden ist, wie in Abb. 2.4 dargestellt.

Statt k-mal zwei Bücher zu vertauschen, verschieben wir dann $(k+1)$-mal ein Buch, was offenbar weniger aufwändig ist, da *einmaliges* Vertauschen *drei* Verschiebeoperationen benötigt. Als Algorithmus sieht das dann so aus:

Bücher sortieren durch Einfügen:

```
1   Gegeben: A: Feld mit n Einträgen;
2   for i := 2 to n do
      // ordne Buch an Position i durch Verschiebungen richtig ein
3     Hand := A[i];      // nimm aktuelles Buch in die Hand
4     j := i − 1;
      // solange korrekte Position des aktuellen Buchs noch nicht erreicht
5     while j ≥ 1 and A[j] > Hand do
6       A[j + 1] := A[j];      // schiebe Buch an Position j nach rechts
7       j := j − 1
8     endwhile
9     A[j] := Hand      // füge aktuelles Buch an der richtigen Stelle ein
10  endfor
```

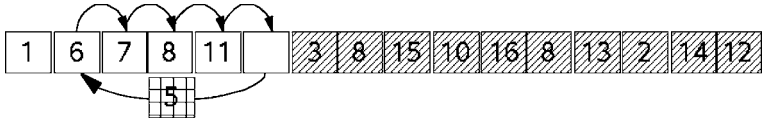

Abb. 2.4. Berechnung der Anzahl der Vertauschungen

Weitere Verbesserungen dieses Sortierverfahrens, wie das gleichzeitige Einfügen von mehr als einem Buch über einen einzelnen Verschiebedurchgang, sowie Animierungen dieser Verfahren findest du auf der folgenden Webseite:

`http://einstein.informatik.uni-oldenburg.de/forschung/animAlgo/`

Vielleicht fragst du dich an dieser Stelle, ob man nicht doch irgendwie die Bücher im Computer gleichzeitig verschieben könnte, um dem aktuell einzusortierenden Buch Platz zu schaffen? Das geht zwar in der realen Welt bis zu einer gewissen Anzahl an Büchern gut, aber der Computer kann so etwas leider nicht auf einmal machen. Allerdings sind in der Vergangenheit spezielle Maschinen entwickelt worden, wie die berühmte Hollerith-Maschine zur Auswertung der amerikanischen Volkszählung, die solche Verschiebungen sozusagen parallel durchführen können. Mehr dazu findest du in Kap. 4.

Auch wenn unser Sortierverfahren also in „normalen“ Computern viel Zeit durch die bis zu $\frac{1}{2}n^2$ Verschiebungen verbrauchen kann, wird er trotzdem gerne verwendet, wenn die Anzahl der zu sortierenden Objekte nicht sehr groß ist oder wenn man damit rechnet, dass die Eingabe „fast“ sortiert ist. Angenehm ist, dass er extrem einfach zu implementieren ist. Im folgenden Kap. 3 werden mit MergeSort und QuickSort wesentlich effizientere Sortierverfahren vorgestellt, die dafür aber etwas schwieriger zu verstehen und zu implementieren sind.

Zum Weiterlesen

1. Das oben vorgestellte Verfahren heißt übrigens auf englisch „Insertion Sort“. Es findet sich in fast jedem Standardwerk über Grundlagen der Algorithmen, beispielsweise ausführlich in: Robert Sedgewick: *Algorithmen in C++*. Pearson Studium, 2002.
2. W.P. Kowalk: *System, Modell, Programm*. Spektrum Akademischer Verlag, 1996 (ISBN 3-8274-0062-7).