Humboldt-Universität zu Berlin
Institut für Informatik

Suchen und Sortieren

Unterrichtsentwurf

Inhalt

1 Einleitung

Such- und Sortierverfahren gehören zu den grundlegenden Anwendungen in der praktischen Informatik und werden laut Rahmenplan bereits im ersten Lernjahr im Lernbereich „Datenbanken und Datenschutz“ behandelt.[1] Für das zweite Lernjahr wird im Abschnitt „Spezielle Algorithmen in typischen Anwendungssituationen“ die Analyse jeweils eines höheren Such- und Sortieralgorithmus vorgeschlagen.[2] Ich gehe davon aus, dass die Beschäftigung mit Such- und Sortierverfahren für die Schülerinnen und Schüler schon länger zurückliegt. Daher habe ich diese Stunde ist als wiederholte Einführung in das Thema „Suchen und sortieren“ geplant. Ich gehe außerdem davon aus, dass die Schülerinnen und Schüler in der Lage sind, einfache Flussdiagramme zu entwerfen.

2 Sachanalyse

Suchverfahren

Das einfachste Suchverfahren ist die lineare Suche. Dieses Suchverfahren ist ein iteratives Verfahren, welches sich auch für die Suche in einer unsortierten Liste eignet. Will man das Vorhandensein eines Elements in einer unsortierten Liste überprüfen bzw. einen dem Suchschlüssel zugeordneten Datensatz auslesen, wird dazu jedes Element der Liste mit dem gesuchten Element verglichen, bis es gefunden, oder sein Nichtvorhandensein in der Liste bestätigt wurde. Diese Suche kann bei großen Datensätzen sehr lange dauern. Im besten Fall wird das gesuchte Element beim ersten Vergleich gefunden, im schlechtesten Fall aber erst, nachdem alle Elemente der Liste mit dem gesuchten Element verglichen wurden. Im Durchschnitt, bei einer Vielzahl von Suchläufen in einer Liste mit n Elementen, wird das gesuchte Element, sofern es in der Liste vorhanden ist, nach n/2 Vergleichen gefunden.[3]
Ein effizienteres Suchverfahren ist die binäre Suche. Voraussetzung für die Anwendung dieses Verfahrens ist das Vorliegen einer sortierten Datensammlung. Für die Bewertung der Effizienz muss der Aufwand für das Sortieren einer unsortierten Liste berücksichtigt werden. Da Datensammlungen im Allgemeinen angelegt werden, um wiederholt auf darin enthaltene Datensätze zugreifen zu können, ist der Aufwand für den einmaligen Sortiervorgang vertretbar, wenn im Anschluss daran eine Vielzahl von Suchläufen durch Anwendung der binären Suche schneller durchgeführt werden können, als dies bei der linearen Suche der Fall wäre.[4] Die binäre Suche ist ein rekursiv implementierbares Verfahren, bei dem der Suchraum nach jedem Vergleich halbiert wird. Dazu wird das gesuchte Element mit dem mittleren Element der Liste verglichen. Stimmen die Elemente überein, ist die Suche damit beendet. Ist dies nicht der Fall, wird die Suche in gleicher Weise in der linken oder der rechten Teilliste fortgesetzt, je nach dem, ob das gesuchte Element kleiner oder größer als das mittlere Element ist. Das Verfahren wird solange fortgesetzt, bis das gesuchte Element gefunden, oder die in Frage kommende Teilliste leer ist. Im besten Fall wird das Element auch hier beim ersten Vergleich gefunden,

1 Vgl. Senatsverwaltung für Bildung, Jugend und Sport Berlin: Curriculare Vorgaben für die gymnasiale Oberstufe. Informatik. Berlin 2005, S. 15.
2 Vgl. ebd., S. 20.
3 Vgl. Sedgewick, Robert: Algorithmen. Bonn; München; Reading Mass. [u. a.] 1991, S. 233 ff.
4 Vgl. ebd., S. 236 ff.

im schlechtesten Fall, bei einer Liste mit n Elementen, nach $\log_2 n$ Vergleichen. Die binäre Suche ist somit insbesondere für die Suche in großen Datensammlungen sehr viel effizienter als die lineare Suche. Bei einer Verdoppelung der Datensammlung wird auch der Aufwand für die lineare Suche verdoppelt, für die binäre Suche wird in diesem Fall nur ein Schritt mehr benötigt.[5]

Sortierverfahren

Hier sollen nur vergleichsbasierte Verfahren betrachtet werden, bei denen die zu sortierenden Elemente paarweise verglichen werden. Die wichtigsten Kenngrößen für die Bewertung von Sortierverfahren sind die Laufzeit, der benötigte Speicherbedarf und die Stabilität des Verfahrens. Ein Verfahren wird als stabil bezeichnet, wenn die Reihenfolge gleicher Elemente einer unsortierten Datensammlung nach dem Sortiervorgang beibehalten wird. Bezüglich des benötigten Speicherplatzes wird im wesentlichen zwischen ortsfesten Verfahren, die keinen oder nur sehr wenig zusätzlichen Speicherplatz benötigen und nicht-ortsfesten Verfahren, die zumindest genügend Speicherplatz benötigen, um eine Kopie der zu sortierenden Liste zu speichern, unterschieden.[6]
Die einfachsten elementaren Sortierverfahren sind Bubblesort, Selectionsort und Insertionsort. Bei Bubblesort wird eine Liste mehrfach vom ersten bis zum letzten Element durchlaufen. Dabei werden immer zwei benachbarte Elemente der Liste miteinander verglichen. Stehen diese nicht in der richtigen Reihenfolge, werden ihre Positionen getauscht. Das Verfahren ist beendet und die Liste sortiert, nachdem ein Durchlauf erfolgt ist, bei dem keine Vertauschungen mehr vorgenommen wurden. Dieses Verfahren ist zwar ortsfest und stabil, benötigt aber für eine Liste mit n Elementen sowohl im Durchschnitt, als auch im schlechtesten Fall, $n^2/2$ Vergleiche und $n^2/2$ Austauschoperationen. Es ist damit vermutlich unter fast allen Umständen nur halb so schnell wie das im Folgenden beschriebene Verfahren Insertionsort.[7]
Bei Insertionsort wird nacheinander jedes Element der unsortierten Liste betrachtet und an der richtigen Stelle innerhalb der bereits betrachteten Elemente eingefügt. Dieses Verfahren ist ebenfalls ortsfest und stabil. Im schlechtesten Fall werden, wie bei Bubblesort, $n^2/2$ Vergleiche, aber nur $n^2/4$ Austauschoperationen benötigt. Im Durchschnitt verringert sich die Zahl der Vergleiche auf $n^2/4$ und die Zahl der Vertauschungen auf $n^2/8$.[8]
Beim letzten der hier betrachteten elementaren Sortierverfahren, Selectionsort, wird der noch unsortierte Teil der Liste linear durchsucht und das kleinste gefundene Element mit dem ersten Element des noch unsortierten Teils der Liste vertauscht. So baut sich innerhalb der Liste von links nach rechts eine aufsteigend sortierte Teilliste auf. Dieses Verfahren ist ortsfest, aber nicht stabil. Für Selectionsort werden in jedem Fall ca. $n^2/2$ Vergleiche und n Austauschoperationen benötigt.[9]
Das wahrscheinlich am häufigsten angewandte höhere Sortierverfahren ist Quicksort. Die Beliebtheit dieses Algorithmus resultiert aus seiner einfachen Implementation, der vielseitigen Verwendbarkeit und dem geringen Ressourcenverbrauch.[10] Quicksort ist ein rekursiver Algorithmus, bei dem die zu sortierende Liste in zwei Teillisten zerlegt wird, die dann unabhängig voneinander nach dem gleichen Prinzip sortiert werden. Für die Zerlegung wird zunächst ein

5 Vgl. Gumm, Heinz-Peter / Sommer, Manfred: Einführung in die Informatik. München; Wien ^{3}1998, S. 255 f.
6 Vgl. Sedgewick 1991, S. 122 f.
7 Vgl. ebd., S. 132 f.
8 Vgl. ebd.
9 Vgl. ebd., S. 131.
10 Vgl. ebd., S. 145.

Vergleichselement gewählt, anhand dessen die zu sortierende Liste so in zwei Teillisten unterteilt wird, dass eine dieser Teillisten alle Elemente enthält, die kleiner sind als das Vergleichselement und die andere Teilliste alle Elemente enthält, die größer sind als das Vergleichselement. Für die Aufteilung der Elemente auf die beiden Teillisten werden vom linken und vom rechten Rand der Liste ausgehend die Elemente paarweise miteinander verglichen. Um die Liste von links nach rechts aufsteigend zu sortieren, werden zwei Elemente dann miteinander vertauscht, wenn das weiter links stehende Element größer und das weiter rechts stehende Element kleiner als das Vergleichselement ist. Wurden auf diese Weise alle Elemente betrachtet, wird das Vergleichselement an die richtige Position gebracht und die beiden Teillisten werden rekursiv nach dem gleichen Prinzip sortiert. Für Teillisten der Länge eins oder null wird die Rekursion abgebrochen. Quicksort ist ortsfest, aber nicht stabil. Wenn man davon ausgeht, dass die zu sortierenden Listen durch geeignete Wahl des jeweiligen Vergleichselements im Durchschnitt in zwei etwa gleich große Teillisten zerlegt werden, benötigt Quicksort für das Sortieren einer Liste mit n Elementen etwa 2n*ln(n) Vergleiche.[11]

3 Stundenplanung

3.1 Ziele

Die Schülerinnen und Schüler sollen am Ende der Stunde in der Lage sein, den prinzipiellen Ablauf der binären Suche zu erklären und diesen Ablauf in Form eines Flussdiagramms zu skizzieren. Sie sollen außerdem die Vor- und Nachteile von linearer und binärer Suche bewerten können. Diesbezüglich sollen sie auch auf die grundsätzlichen Einsatzmöglichkeiten hinsichtlich sortierter/unsortierter Listen eingehen und zum Aufwand für das Sortieren einer unsortierten Liste, als Vorbedingung für die Anwendung der binären Suche, Stellung nehmen können.
Zur Vorbereitung auf die nächste Stunde soll mindestens ein elementares Sortierverfahren erklärt werden. Dadurch habe ich zu Beginn der nächsten Stunde die Möglichkeit, nach kurzer Rekapitulation dieses Verfahrens, zügig in die Erarbeitung des höheren Sortierverfahrens Quicksort einzusteigen. Im Rahmen des Einstiegs sollen dazu die animierten Abläufe verschiedener Sortierverfahren demonstriert, und hinsichtlich ihrer Laufzeitunterschiede bewertet werden.

3.2 Didaktische Analyse

Zu Beginn des Unterrichts erkläre ich in kurzen Worten den Ablauf der Stunde. Ich möchte in dieser Stunde zunächst die lineare und die binäre Suche behandeln. Dabei liegt der Schwerpunkt auf der Betrachtung der einzelnen Teilschritte der binären Suche. Die unterschiedliche Effizienz der Verfahren soll durch zwei, von den Schülerinnen und Schülern durchzuführende Suchen, praktisch erfahrbar werden. Ich zeige zunächst mit dem Overhead-Projektor eine unsortierte Liste.[12] Diese enthält Namen, und den Namen zugeordnete Telefonnummern. Ich er-

11 Vgl. Sedgewick 1991, S. 146 f. u. 151.
12 Siehe Anhang, S. 10.

kläre, dass dies eine typische Liste ist, die man z. B. als Mitglied eines Vereins bekommt und auf der die Vereinsmitglieder und ihre Telefonnummern aufgelistet sind. Ich bitte dann die Schülerinnen und Schüler, die Telefonnummer zu dem Namen Schröder herauszusuchen und die Nummer zu nennen. Dann bitte ich auch noch darum, die Telefonnummer von Krause herauszusuchen. Ich habe absichtlich zwei Namen gewählt, die im letzten Drittel der Liste liegen, denn Sinn und Zweck dieser Übung ist es, den Nachteil der linearen Suche praktisch zu verdeutlichen. Die Darbietung der Liste mit dem Overhead-Projektor habe ich bewusst gewählt, um das Suchproblem plausibler zu machen. Bei einer Darbietung der Liste mit dem Beamer läge automatisch die Benutzung der Suchfunktion des Editors nahe. Dann frage ich die Schülerinnen und Schüler, wie sie bei der Suche nach den Namen vorgegangen sind. Die Antwort wird entweder von den Schülerinnen und Schülern selbst, oder abschließend von mir so formuliert, dass das Prinzip der linearen Suche deutlich wird. Ich sage, dass dieses Vorgehen zwar unbequem ist, bei einer kurzen Liste wie dieser aber noch ohne größere Probleme angewendet werden kann. Dann frage ich, wie die Suche bei einer sehr langen Liste vor sich gehen könnte. Ich gehe davon aus, dass aus der Klasse der Vorschlag kommt, dass eine sehr lange Liste auf jeden Fall alphabetisch sortiert sein müsste. Ich hole dann ein Telefonbuch hervor und sage, dass ich eine sehr lange, alphabetisch sortierte Liste mitgebracht habe und bitte einen Schüler oder eine Schülerin nach vorn. Dieser oder diese soll dann die Telefonnummer zum ersten Eintrag mit dem Namen Schröder suchen. An die Klasse richte ich den Auftrag, den Schüler oder die Schülerin zu beobachten und hinterher den Ablauf der Suche zu beschreiben. Falls beim ersten Aufschlagen des Telefonbuchs zufällig die richtige Seite aufgeschlagen wurde, bzw. die Suche nur sehr kurz dauerte, kann noch eine zweite Suche nach dem ersten Eintrag zum Namen Krause folgen. Dann frage ich die Schülerinnen und Schüler nach ihren Beobachtungen. Möglicherweise wird der prinzipielle Ablauf der binären Suche bereits formuliert, falls nicht, werde ich das tun. Ich mache dabei noch einmal die Unterschiede der beiden Suchverfahren deutlich.

Nun folgt die Arbeitsphase, die ich an dieser Stelle geplant habe, um den Schülerinnen und Schülern die Gelegenheit zu geben, das Verständnis der binären Suche zu vertiefen. Zu diesem Zweck sollen die Schülerinnen und Schüler versuchen, den Ablauf der Suche in Form eines Flussdiagramms darzustellen. Um den Entwurf des Diagramms zu unterstützen, habe ich eine Aufgabe vorangestellt, die den Ablauf der Suche noch einmal veranschaulichen soll.[13] Zu Beginn der nun folgenden Arbeitsphase gebe ich das Aufgabenblatt aus und erkläre noch einmal mündlich den Arbeitsauftrag. Sicherheitshalber frage ich nach den typischen Symbolen eines Flussdiagramms, die ich dann an der Tafel skizziere. Während der Arbeitsphase stehe ich als Ansprechpartner für eventuelle Verständnisfragen zur Verfügung. Im Anschluss an die Arbeitsphase soll zunächst die erste Aufgabe besprochen werden.[14] Ich werde dazu einen Schüler oder eine Schülerin bitten, nur kurz die einzelnen Suchschritte durchzugehen. Dann bitte ich einen Schüler oder eine Schülerin das Flussdiagramm an der Tafel vorzustellen. Im Laufe der Vorstellung achte ich darauf, dass alle wesentlichen Punkte berücksichtigt werden.[15]

Da die binäre Suche das Vorhandensein einer sortierten Liste voraussetzt, stellt sich an dieser Stelle die Frage, auf welche Weise eine Liste sortiert werden kann. Um das Prinzip eines elementaren Sortierverfahrens mit den Schülerinnen und Schülern zu erarbeiten, zeichne ich eine Liste mit neun Feldern an die Tafel und befestige mit Magneten in jedem Feld einen Zettel mit

13 Siehe Anhang, S. 11.
Dieses Aufgabenblatt habe ich in Anlehnung an eine Aufgabenstellung von Siegfried Spolwig entworfen. Vgl. Spolwig, Siegfried: Suchen und Sortieren. Rekursive binäre Suche. 2006. URL: http://oszhdl.be.schule.de/gymnasium/faecher/informatik/sort-such/suchen_bin21.htm. Stand: 22.07.07.

14 Siehe Anhang, S. 12.

15 Siehe Anhang S. 13.

einer Zahl. Zur Vereinfachung des nun folgenden Sortiervorgangs habe ich neun verschiedene Zahlen gewählt. Die Zahlen stehen in ungeordneter Reihenfolge an der Tafel. Dann bitte ich um Vorschläge, wie diese Liste sortiert werden kann. Als Bedingung gebe ich an, dass keine neue Liste eröffnet und die Zahlen innerhalb der Liste aufsteigend sortiert werden sollen. Bei dieser Aufgabenstellung liegt sicherlich ein Sortierverfahren wie Selctionsort oder Bubblesort nahe. Ich gehe zwar davon aus, dass die Vorschläge der Schülerinnen und Schüler auf das Verfahren Selectionsort hinauslaufen werden, stelle mich aber auch darauf ein, dass möglicherweise ungewöhnliche Vorschläge geäußert werden, die ich in der Klasse zur Diskussion stellen werde. In jedem Fall werde ich den Ablauf von Selectionsort auch in wenigen Stichworten an der Tafel notieren und die Schülerinnen und Schüler bitten, dies in ihre Aufzeichnungen zu übernehmen. Falls die Zeit noch reicht oder die Vorschläge der Schülerinnen und Schüler dies anbieten, notiere ich auch den Ablauf von Bubblesort. Am Schluss gebe ich einen kurzen Ausblick auf den Inhalt der nächsten Stunde.

3.3 Verlaufsplan

Zeit (min)	didaktische Funktion und Methoden	Lehreraktivität	Schüleraktivität	Medien
10	**Einstieg** Praktische Beispiele	Ich erkläre zuerst den Ablauf der Stunde.		
	1. Telefonnummer in unsortierter Liste	Ich zeige eine unsortierte Namensliste und frage nach zwei Telefonnummern.	Schülerinnen und Schüler suchen Telefonnummern in der Liste.	Overhead-Projektor
	2. Telefonnummer im Telefonbuch	Ich frage nach einer Telefonnummer aus dem Telefonbuch.	Eine Schülerin oder ein Schüler sucht die Nummer. Die anderen beobachten den Ablauf.	Telefon-buch
	Lehrer-Schüler-Gespräch	Besprechung der beiden Suchverfahren.	Beteiligen sich am Gespräch.	
15	**Arbeitsphase** Einzelarbeit	Ich erkläre den Arbeits-auftrag und stehe als Ansprechpartner zur Verfügung.	Die Schülerinnen und Schüler bearbeiten das Aufgabenblatt.	Aufgaben-blatt
10	**Ergebnissicherung** Besprechung der Ergebnisse	Ich bitte einen Schüler oder eine Schülerin an die Tafel.	Eine Schülerin oder ein Schüler stellt das Flussdiagramm an der Tafel vor.	Tafel
10	**Erarbeitung** Elementare Sortierverfahren	Ich moderiere die Erarbeitung mindestens eines elementaren Sortierverfahrens.	Die Schülerinnen und Schüler beteiligen sich an der Erarbeitung.	Tafelappli-kation

4 Quellenverzeichnis

Gumm, Heinz-Peter / Sommer, Manfred: Einführung in die Informatik. München; Wien [3]1998.

Sedgewick, Robert: Algorithmen. Bonn; München; Reading Mass. [u. a.] 1991.

Senatsverwaltung für Bildung, Jugend und Sport Berlin: Curriculare Vorgaben für die gymnasiale Oberstufe. Informatik. Berlin 2005.

Spolwig, Siegfried: Suchen und Sortieren. Rekursive binäre Suche. 2006. URL: http://oszhdl.be.schule.de/gymnasium/faecher/informatik/sort-such/suchen_bin21.htm. Stand: 22.07.07.

5 Anhang

Folie mit unsortierter Liste

1. Müller: 9324687
2. Schmidt: 5676535
3. Schneider:65461
4. Fischer: 65468465
5. Weber: 446133
6. Meyer: 51798465
7. Wagner: 6546154
8. Becker: 5654651
9. Schulz: 65468465
10. Hoffmann: 654651654
11. Schäfer: 9465651
12. Koch: 6675618
13. Bauer: 15465581
14. Richter: 1284524
15. Klein: 71655524
16. Wolf: 455845481
17. Schmitt: 18974651
18. Neumann: 218452418
19. Schwarz:168417545
20. Zimmermann: 18451541
21. Braun: 216845841
22. Krüger: 46127415
23. Hofmann: 1545552
24. Hartmann: 21981415
25. Lange: 1541324651
26. Schröder: 218541541
27. Werner: 19846511
28. Schmitz: 8994615
29. Krause: 189454145
30. Meier: 184651115

Aufgabenblatt zur binären Suche (unausgefüllt)

Binäre Suche

Aufgabe 1:

Finde in der unten stehenden Liste den Buchstaben S mit Hilfe der binären Suche. Kreise dazu das mittlere Element ein (gibt es kein einzelnes mittleres Element, dann wähle eines der beiden, die der Mitte am nächsten sind) und übertrage nur die Elemente in die nächste Zeile, die beim folgenden Suchlauf noch berücksichtigt werden müssen.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1.	A	C	E	G	H	K	L	M	O	P	Q	S	T	V	W	X	Y
2.																	
3.																	
4.																	
5.																	
6.																	

Aufgabe 2:

Beschreibe den Programmablauf der binären Suche in Form eines Flussdiagramms. Das Programm soll feststellen, ob sich ein gesuchtes Element in einer aufsteigend sortierten Liste befindet.

Aufgabenblatt zur binären Suche – Beispiellösung zur Aufgabe 1

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1.	A	C	E	G	H	K	L	M	**[O]**	P	Q	S	T	V	W	X	Y
2.										P	Q	S	**[T]**	V	W	X	Y
3.										P	**[Q]**	S					
4.												**[S]**					
5.																	
6.																	

Aufgabenblatt zur binären Suche – Beispiellösung zur Aufgabe 2