Proseminar „Pioniere der Informatik“

Rudolf Bayer und B-Bäume

Alexander Regler

Technische Universität München
regleral@in.tum.de

Abstract: Diese Arbeit gibt zunächst einen Überblick über den Werdegang und das Schaffen von Rudolf Bayer, einem Pionier der Informatik, insbesondere im Bereich der Datenbanken. Darüber hinaus wird seine bedeutendste Erfindung vorgestellt: Die B-Bäume, welche aufgrund ihrer Effizienz in fast allen modernen relationalen Datenbanken als Datenstruktur zur Organisation und Verwaltung von großen Datenmengen eingesetzt werden.

1 Einleitung

Gegen Ende der 60er Jahre suchte man unter anderem für die Fertigung von Flugzeugen, die wie der Jumbo Jet aus bis zu 500.000 Einzelteilen bestanden, effektive Produktionsverfahren. Auf der anderen Seite präsentierte IBM eine Plattenspeichertechnologie, die die Speicherung von größeren Datenmengen und schnellere Zugriffszeiten ermöglichte. Aufgrund dieser beiden Entwicklungen hatte man „die Vision, interaktive Informationssysteme realisieren zu können“. Die fundamentale Fragestellung lautete daher: „Wie organisiert man größere Datenmengen auf dieser Plattentechnologie?“ [SP02]

Die bis dahin bekannten Binärbäume[1] bargen den entscheidenden Nachteil, dass sie entarten können (siehe Abbildung 1), wodurch die Suche nach einem Baumelement im schlimmsten Fall nur in linearer Zeit[2] durchgeführt werden kann. Ein regelmäßiges Neusortieren des Baumes zur Beseitigung der Entartung ist ebenfalls ineffizient [LT03].

[1] Binärbaum: Gewurzelter Baum, in dem jeder Baumknoten höchstens 2 Kindknoten hat, wobei der Schlüssel des linken Kindknotens kleiner und der Schlüssel des rechten Kindknotens größer als der des Vaterknotens ist.
[2] Bei einer Suche in linearer Zeit müssen im schlimmsten Fall alle Baumelemente durchsucht werden. Beispielsweise werden in Baum B aus Abbildung 1 bei der Suche nach dem Element 5 alle Elemente betrachtet.

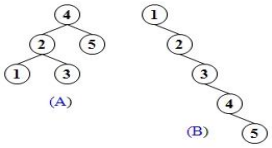

Abbildung 1: Binärbaum – nicht entartet (A) und entartet (B)

Rudolf Bayer hatte 1969 die Lösung für die damalige Problemstellung: Zur Organisation und Verwaltung sehr großer Datenmengen erfand er eine baumähnliche Datenstruktur, welche sich beim Einfügen oder Löschen von Baumelementen selbst ausbalanciert und damit eine Entartung vermieden wird: Die B-Bäume [IN09][LT03].

In dieser Arbeit wird im 2. Kapitel zunächst der Erfinder der B-Bäume vorgestellt. Kapitel 2.1 schildert Rudolf Bayers Lebenslauf, Kapitel 2.2 seine Entwicklungs- und Forschungstätigkeiten im Bereich der Informatik und Kapitel 2.3 seine dafür erhaltenen Ehrungen und Auszeichnungen.

Das 3. Kapitel beleuchtet den B-Baum selbst: Seine Eigenschaften (Kapitel 3.1) und seine Basisoperationen (Kapitel 3.2) des Suchens nach Elementen, des Einfügens und des Löschens von Elementen. Jede Operation wird zunächst allgemein erklärt und an jeweils mindestens einem Beispiel demonstriert.

Im Schlussteil wird zum einen auf die Effizienz und den Stellenwert der B-Bäume verwiesen und zum anderen Rudolf Bayer eine der ältesten offenen Fragen der Informatik gestellt.

Im nun folgenden Kapitel wird zunächst die Person Rudolf Bayer vorgestellt.

2 Rudolf Bayer

2.1 Lebenslauf

Professor Rudolf Bayer, Ph.D. wurde am 3. März 1939 im unterfränkischen Greßthal (Gemeinde Wasserlosen) geboren, wo er seine Kindheit verbrachte. Als 9- bis 11-jähriges Kind begeisterte er sich dafür, seinem Vater, der Dorfschmied und Landmaschinenmeister war, bei dessen Arbeit zu helfen, beispielsweise beim Schmieden von Wagennägeln per Hand. Mit 11 Jahren durfte er das Realgymnasium in Würzburg besuchen, welches ihm noch mehr Freude bereitete [LE03][IN09][IN10].

1959 nahm Rudolf Bayer das Studium der Mathematik an der Technischen Universität München auf, an der man sich bereits intensiv mit Informatikthemen wie Perm oder Algol 60 Compiler beschäftigte. Ein eigenständiger Studiengang der Informatik existierte noch nicht. 1962 legte er sein Vordiplom ab [CV09][IN09].

Ein Jahr später wechselte er von der Technischen Universität München zur University of Illinois (USA), an der er die akademischen Grade Master of Science im Jahre 1964 und Ph.D.[3] im Jahre 1966 erwarb. Im Sommersemester 1966 arbeitete Rudolf Bayer als „Assistant Professor of Computer Science“ an der University of Illinois. Von 1966 bis 1970 beschäftigten ihn Boeing Scientific Research Laboratories in Seattle als Forscher. Anschließend war er als „Associate Professor“ des Computer Sciences Department der Purdue University in Lafayette (Indiana, USA) bis 1972 tätig [CV09].

Abbildung 2: Prof. Rudolf Bayer, Ph.D. [SI01]

1972 wurde er als Professor an die Technische Universität München berufen, an der er bis 2004 den Lehrstuhl III (zuletzt „Datenbanken und Wissensbasen“) innehatte. In dieser Zeit war er zudem als Gastprofessor von 1975 bis 1976 im IBM San Jose Research Center (Kalifornien, USA), 1985 im XEROX Palo Alto Research Center (Kalifornien, USA), 1994 an der University of Library Sciences in Tsukuba (Japan), 1995 an der La Trobe University in Melbourne (Australien) und 2000 an der National University of Singapore tätig [CV09].

Neben seinen Tätigkeiten als Professor übte Rudolf Bayer aber auch verschiedenste weitere Beschäftigungen aus, von denen hier nur ein paar erwähnt seien: Als Berater unterstützte er unter anderem IBM, Siemens, Amdahl, DEC, Data General und die Deutsche Telekom. 1987 war er Mitbegründer der Datenbankfirma TransAction Software GmbH und ist seitdem ihr Aufsichtsratsvorsitzender. Seit 1988 leitet er die Research Group Knowledge Bases am Bavarian Research Center for Knowledge Based Systems (FORWISS) [CV09][LE03][IN09].

Im Folgenden wird nun sein Beitrag im Bereich der Informatik, insbesondere die Erfindung der B-Bäume, die ihn zu einem Pionier der Informatik machten, beschrieben.

2.2 Bayers Wirken in der Informatik

2.2.1 Entwicklung der B-Bäume

Rudolf Bayer erfand und gemeinsam mit seinem Kollegen Dr. Edward M. McCreight entwickelte er im Herbst 1969 das Konzept der B-Bäume, welche als Datenstruktur zur Organisation und zur Verwaltung sehr großer Datenmengen verwendet werden [SP02].

[3] Ph.D. (doctor of philosophy): Doktortitel, den man unter anderem in den USA erwerben kann. Voraussetzung ist ein abgeschlossenes Masterstudium.

Abbildung 3: Dr. Edward M. McCreight [MC08]

Aufgrund einer Speicherausnutzung[4] von lediglich 50 % im schlechtesten Fall und des neuartigen Wachstums eines B-Baumes[5] wurde ihr gemeinsames Paper, welches sie an der damals renommiertesten Zeitschrift („Communications of the ACM“) einreichten, zurückgewiesen. Erst im Jahre 1972, mit Unterstützung von Prof. Donald E. Knuth[6], der das Potential der Arbeit erkannte, wurden die B-Bäume von Acta Informatica akzeptiert und publiziert [SP02].

Ebenfalls im Jahre 1969 konzipierte Edgar F. Codd das theoretische Modell der relationalen Datenbanken[7], welches 1970 publiziert wurde [SP02].

Beide Theorien, die der relationalen Datenbanken und die der B-Bäume, wurden von 1975 bis 1976 im Datenbanksystem „System R“ durch IBM in Forschungsprojekten realisiert, an dem Rudolf Bayer in seinem Forschungsjahr in San Jose direkt beteiligt war [FS07][SP02]. Hierbei wurde unter Mitwirkung von Rudolf Bayer ein Synchronisations-protokoll zur Ausführung von parallelen Datenbankanfragen entwickelt, welches die erfolgreiche Integration der B-Bäume in das Datenbanksystem „System R“ ermöglichte [FS07].

Die unter anderem durch B-Bäume effizienten relationalen Datenbanksysteme setzten sich gegenüber anderen Datenbanksystemen durch, so dass heutzutage nahezu alle kommerziellen Datenbanksysteme die von Rudolf Bayer entwickelten B-Bäume verwenden [FS07].

Zweifelsfrei ist das Konzept der B-Bäume bis heute die größte Erfindung Rudolf Bayers. Bevor dieses Konzept in Kapitel 3 ausführlich betrachtet wird, werden zunächst weitere Verdienste Rudolf Bayers im Bereich der Informatik gezeigt.

---

[4] Speicherausnutzung: Anteil des tatsächlich mit Daten belegten Speicherplatzes zum gesamten reservierten Speicherplatz.

[5] Wachstum eines B-Baumes: Der B-Baum wächst im Gegensatz zu den meisten Bäumen in der Natur oder den bis 1969 in der Informatik verwendeten Bäumen nicht an seinen Blättern, sondern an seiner Wurzel.

[6] Prof. Donald E. Knuth: Professor für Informatik an der Stanford University seit 1968. Er wurde für seine Verdienste in der Informatik unter anderem mit dem Turing-Award ausgezeichnet. „Donald Knuth hat die Etablierung der Informatik als eigenständige Wissenschaft […] wie kaum ein Anderer vorangetrieben“ [DK05].

[7] relationales Datenbankenmodell: Modell zur Speicherung von Daten in zweidimensionalen Tabellen (Entitäten), wobei eine Zeile genau ein Element dieser Entität definiert und die Spalten unterschiedliche Eigenschaften dieses Elements beschreiben. Beispielsweise könnte man eine Entität Professoren mit Spalten für Name, Universität und Geburtsjahr festlegen, dann wäre [Rudolf Bayer, Technische Universität München, 1939] als ein Element dieser Entität in einer Zeile dieser Tabelle abgelegt.

2.2.2 Weitere Entwicklungs- und Forschungstätigkeiten

Rudolf Bayers weitere Entwicklungs- und Forschungstätigkeiten in der Informatik können wie die B-Bäume nahezu gänzlich dem Gebiet der Datenbanken zugeordnet werden, mit denen er sich seit 1966, unmittelbar nach seiner Promotion, beschäftigt [IN09].

Er erforschte und entwickelte effektive Methoden für synchronisierte oder parallele Transaktionen sowohl in zentralen wie auch in verteilten Datenbanksystemen. In den Bereichen deduktiver Datenbanken[8], objektorientierter Datenbanken[9], wissensbasierter Systeme[10] sowie für Digitalbibliotheken veröffentlichte er eine Vielzahl von Publikationen, die effiziente Realisierungen ermöglichten. Darüber hinaus beschäftigte er sich mit data warehousing[11] und dem Datenschutz [CV09][PU08].

Zudem entwickelte Rudolf Bayer Präfix-B-Bäume, welche eine Weiterentwicklung der B-Bäume darstellen und deren Performanz erhöhen, sowie UB-Bäume. Während B-Bäume den Zugriff bei eindimensionalen Datenbankanfragen beschleunigen, eignen sich UB-Bäume als Datenstruktur zur Verwendung für mehrdimensionale Bereichsanfragen[12] [CV09][SP02].

In einem anderen Bereich der Informatik entwickelte er einen Algol 60 Compiler[13] in seiner Zeit an der Technischen Universität München und an der University of Illinois. Hierzu erstellte Rudolf Bayer 1967 auch den „wohl erste[n] ernsthafte[n] symbolische[n] Debugger, welcher die Fehleranalyse auf der Ebene des Quellcodes lieferte“ [IN09].

Aufgrund der genannten Entwicklungen, aber vor allem aufgrund der Erfindung des B-Baumes, wurde Rudolf Bayer mehrfach geehrt.

2.3 Ehrungen und Auszeichnungen

Rudolf Bayer wurde 1999 mit dem Bundesverdienstkreuz 1. Klasse ausgezeichnet [DS99].

2001 wurde ihm speziell für seine Erfindung der B-Bäume der ACM SIGMOD Innovation Award verliehen, der eine „außerordentliche Auszeichnung […] für innovative Beiträge von hoher Bedeutung und anhaltendem Wert für die Entwicklung, das Verstehen oder die Nutzung von Datenbanksystemen“ darstellt [DS01].

---

[8] deduktive Datenbank: Erweiterung einer relationalen Datenbank um logische Regeln, aus denen zusätzliches Wissen gewonnen werden kann [DD06].
[9] objektorientierte Datenbank: Daten werden als Objekte in der Datenbank gespeichert [OD09].
[10] wissensbasiertes System: Computerprogramm, das mit Hilfe von formalisiertem Wissen über einen bestimmten Anwendungsbereich Informationen auswertet [WS09].
[11] data warehousing: „Datenbasis für die Analyse und Entscheidungsunterstützung für das Management“ [DW09].
[12] mehrdimensionale Bereichsanfrage: Selektion von Elementen einer Datentabelle unter Berücksichtigung von mindestens 2 Eigenschaften.
[13] Compiler: Computerprogramm, das einen in einer Programmiersprache verfassten Code in semantisch äquivalente Maschinenanweisungen zur Ausführung des Computers umwandelt.

Im Jahre 2005 wurde Rudolf Bayer als „herausragender Hochschullehrer, Wissenschaftler und Erfinder“ [GI05] von der Gesellschaft für Informatik (GI) geehrt und zum Fellow der GI ernannt. Hiermit wurden seine Erfindungen und Veröffentlichungen grundlegender Konzepte für Datenbanksysteme gewürdigt, die „schon sehr früh und weltweit Eingang in Produkte gefunden“ [GI05] haben [DS05][GI05][JF05].

Abbildung 4: Ernennung zum Fellow der GI [GI05]

Die B-Bäume als das wohl bedeutendste grundlegende Konzept für Datenbanksysteme, welches Rudolf Bayer erfunden hat, werden im Folgenden vorgestellt.

3 B-Bäume

3.1 Definition und Eigenschaften

B-Bäume sind eine Datenstruktur zur Organisation und Verwaltung von Daten. Man kann Daten über diese Struktur suchen, einfügen oder löschen. Zunächst seien aber die grundlegenden Eigenschaften von B-Bäumen erwähnt, bevor in Kapitel 3.2 die einzelnen Operationen erörtert werden.

Ein B-Baum besteht aus Knoten, in denen Einträge[14] E_i und Verweise V_i auf weitere Knoten gespeichert sind. Ein Eintrag E_i mit Schlüssel[15] S_i ist in einem Knoten stets nach einem Verweis V_{i-1} und vor einem Verweis V_i gespeichert. Hierbei verweist V_{i-1} auf einen Knoten, der nur Schlüssel beinhaltet, die kleiner als S_i sind, wohingegen V_i auf einen Knoten mit Schlüsseln verweist, die allesamt größer als S_i sind[16]. In Abbildung 5 ist ein Knoten eines B-Baumes mit 3 Einträgen und einem freien Bereich, in dem noch weitere Einträge gespeichert werden könnten, dargestellt [DB06].

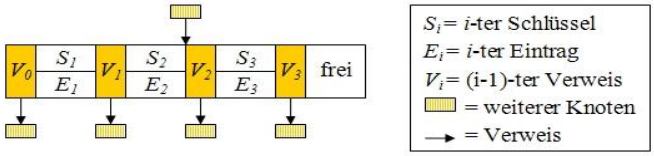

Abbildung 5: Allgemeine Form eines Knotens im Baum

[14] Eintrag: Ein Eintrag ist entweder ein Datensatz oder ein Verweis auf einen Datensatz. Verweise werden bevorzugt, wenn der referenzierte Datensatz zu groß für die Speicherung in einem Knoten des Baumes ist.
[15] Schlüssel: Jeder Eintrag besitzt einen einzigartigen Schlüssel. Die Schlüssel lassen sich ihrem Wert entsprechend ordnen, so dass je 2 Schlüssel unterschieden werden können.
[16] Handelt es sich um einen Blattknoten, so sind die Verweise nicht definiert.

Aufgrund der Verweise eines Knotens auf andere Knoten lassen sich die Knoten in einer baumähnlichen Struktur anordnen. Hierbei wird der oberste Knoten als Wurzel bezeichnet, die Knoten auf der untersten Ebene als Blätter bzw. Blattknoten. Der Beispielbaum in Abbildung 6 besteht aus der Wurzel A und aus den 6 Blattknoten D bis I [DB06].

Nach [DB06] ist die Struktur eines B-Baumes mit Grad *k* definiert über die folgenden Eigenschaften:

- Der B-Baum ist balanciert, das heißt, jeder Weg von dem Wurzelknoten zu einem Blatt über die Verweise der dazwischen liegenden Knoten hat die gleiche Länge.
- Der Wurzelknoten hat zwischen einem und 2*k* Einträgen. Alle weiteren Knoten haben mindestens *k* und höchstens 2*k* Einträge. Die Einträge werden entsprechend ihrer Schlüsselwerte in allen Knoten sortiert gehalten.
- Alle Knoten mit *n* Einträgen haben *n*+1 Kindknoten. Die Blätter besitzen keine Kinder, ihre Verweise sind nicht definiert.
- Gegeben sei ein beliebiger Knoten (kein Blattknoten) mit Schlüsseln S_1, S_2, ..., S_n und den Verweisen V_0, V_1, ..., V_n auf dessen *n*+1 Kindknoten. Dann verweist V_0 auf einen Teilbaum mit Schlüsseln kleiner als S_1, V_i (mit i = 1, 2, …, *n*-1) weist auf einen Teilbaum, dessen Schlüssel zwischen S_i und S_{i+1} liegen und V_n weist auf einen Teilbaum mit Schlüsseln größer als S_n.

Ein Beispiel für einen B-Baum des Grades *k* = 2 findet sich in Abbildung 6. Die Verweise zwischen Knoten werden als Pfeile dargestellt. Auf Einträge wurde zur Erhöhung der Übersichtlichkeit verzichtet.

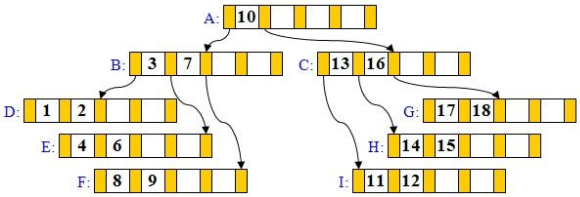

Abbildung 6: Beispielbaum (Grad k = 2)

Nachdem die Eigenschaften eines B-Baumes definiert wurden, werden im Folgenden die Operationen zum Suchen, Einfügen oder Löschen von Einträgen, welche die B-Baum-Eigenschaften nicht verletzen, vorgestellt.

3.2 Operationen auf B-Bäumen

3.2.1 Das Suchen

Das Suchen eines Eintrages *E* in einem B-Baum erfolgt über seinen Schlüssel *S*, der vorgegeben ist.

Begonnen wird die Suche an der Wurzel des Baumes. Die bereits in sortierter Reihenfolge angeordneten Schlüssel in diesem Knoten (S_1, S_2, …, S_n) werden mit dem Schlüssel S verglichen [IN10].

Entspricht ein Schlüssel S_i dem Schlüssel S, so kann direkt auf dessen Eintrag E_i zugegriffen werden.

Andernfalls befindet sich der Eintrag E nicht in diesem Knoten. In diesem Fall wird als Nächstes dem Verweis V_i gefolgt, dessen nachfolgender Schlüssel S_{i+1} größer und dessen vorausgegangener Schlüssel S_i kleiner als der gesuchte Schlüssel S ist (vgl. Abbildung 5). Ist S_1 größer bzw. S_n kleiner als S, so wird direkt dem ersten Verweis V_0 bzw. dem letzten Verweis V_n zum nächsten Knoten gefolgt.

Die Schlüssel dieses erreichten Knotens werden wieder mit S verglichen, der gesuchte Eintrag in diesem Knoten gefunden oder dem entsprechenden Verweis analog zum eben erläuterten Vorgehen gefolgt. Dies geschieht solange, bis der gesuchte Eintrag gefunden oder ein Blattknoten erreicht wurde, der den gesuchten Eintrag nicht enthält. In letzterem Fall ist durch die Eigenschaften des B-Baumes garantiert, dass der Eintrag auch in keinem nicht durchsuchten Knoten des Baumes gespeichert ist.

Beispielsweise möchte man den Eintrag mit Schlüssel $S = 4$ in dem Baum aus Abbildung 6 finden. Zunächst wird in der Wurzel S mit 10 verglichen. Da $S = 4$ kleiner als 10 ist, wird dem 1. Verweis zu Knoten B gefolgt. Da S größer als 3 und kleiner als 7 ist, wird anschließend der Knoten E betrachtet. Dessen 1. Schlüssel stimmt mit S überein, so dass sich der gesuchte Eintrag ebenfalls an der 1. Position für einen Eintrag im Knoten E befinden muss.

3.2.2 Das Einfügen

Beim Einfügen eines Eintrages mit Schlüssel S, welcher zuvor nicht im Baum enthalten ist, wird zunächst seine Einfügestelle gesucht. Dies erfolgt gemäß des in Kapitel 3.2.1 beschriebenen Absteigens im Baum, bis ein Blattknoten erreicht ist [DB06].

In diesen Blattknoten wird der Schlüssel an der Einfügestelle eingefügt [DB06].

Besitzt der Knoten, in den der Eintrag hinzugefügt werden soll, damit mehr als $2k$ Einträge, so muss er geteilt werden: Es wird ein neuer Knoten erzeugt, in den alle Einträge, deren Schlüssel größer sind als der Schlüssel des mittleren Eintrages, eingefügt werden. Der mittlere Eintrag wird in den Vaterknoten, unter Beibehaltung der Sortierung der Schlüssel in diesem Knoten, aufgenommen. Der im Vaterknoten unmittelbar nachfolgende Verweis wird mit dem neuen Knoten verbunden [DB06].

Sollte nun der Vaterknoten mehr als $2k$ Elemente beinhalten, so muss auch dieser analog des vorangegangenen Absatzes aufgeteilt werden. Handelt es sich außerdem bei dem zu teilenden Vaterknoten um die Wurzel des Baumes, so muss eine neue Wurzel, lediglich aus dem mittleren Eintrag bestehend, erzeugt werden [DB06].

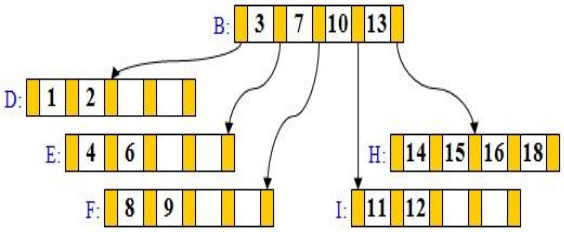

Abbildung 7: Beispielbaum vor dem Einfügen der 17 (Grad k = 2)

Das Einfügen eines Eintrages in einen Knoten, der weniger als 2*k* Einträge hat, ist also ohne das Spalten dieses Knotens möglich. Beispielsweise würde bei dem in Abbildung 7 dargestellten Baum der Eintrag mit dem Schlüssel 5 nach der Suche der Einfügestelle in dem Blattknoten E zwischen den Schlüsseln 4 und 6 hinzugefügt werden.

Dagegen gestaltet sich in dem Beispielbaum in Abbildung 7 das Einfügen eines Eintrages mit Schlüssel 17 komplizierter, da der Knoten H bereits 2*k* Einträge besitzt und dadurch aufgespaltet werden muss. Hierbei wird der mittlere Eintrag mit Schlüssel 16 in den Vaterknoten B verschoben. Für die Einträge mit größerem Schlüssel als 16, nämlich 17 und 18, wird ein neuer Knoten (Knoten G) angelegt. Außerdem wird der Verweis rechts des Schlüssels 16 in Knoten B mit einer Referenz auf Knoten G definiert. Allerdings besitzt nun Knoten B 5 Einträge und damit mehr als 2*k*, weshalb auch dieser aufgespaltet werden muss: Aus dem mittleren Eintrag mit Schlüssel 10 wird der Wurzelknoten A gebildet, dessen 2. Verweis auf den neu zu schaffenden Knoten C mit den Schlüsseln 13 und 16 zeigt. Nun haben alle Knoten außer der Wurzel zwischen *k* und 2*k* Einträge und die Einfügeoperation ist beendet. Entstanden ist der B-Baum aus Abbildung 6, dessen Höhe durch das Einfügen einer neuen Wurzel um eins gewachsen ist.

3.2.3 Das Löschen

Beim Löschen eines Eintrages mit Schlüssel *S* in einem B-Baum wird zunächst die Position des Eintrages durch die Suchoperation (Kapitel 3.2.1) ermittelt [DB06].

Befindet sich der zu löschende Eintrag in einem Blattknoten, so wird dieser Eintrag gelöscht. Ist der Eintrag hingegen in einem inneren Knoten[17] gespeichert, so wird der zu löschende Eintrag mit dem Eintrag des nächstgrößeren (oder nächstkleineren) Schlüssels ersetzt, damit die Verweise zu den Kindknoten erhalten bleiben [DB06].

In beiden Fällen kann es zu einer Unterbesetzung[18] eines Blattknotens kommen, die zur Aufrechterhaltung der Eigenschaften des B-Baumes gelöst werden muss [DB06].

[17] innerer Knoten: Knoten des Baumes, aber kein Blattknoten.
[18] Unterbesetzung: Ein Knoten weist weniger als *k* Einträge auf.

Der unterbesetzte Blattknoten kann mit einem Eintrag aus einem Nachbarknoten[19] aufgefüllt werden, sofern der Nachbarknoten mehr als *k* Einträge besitzt. Hierbei wird von dem linken (bzw. rechten) Nachbarknoten der letzte (bzw. erste) Eintrag in den Vaterknoten an Stelle des Eintrages links (bzw. rechts) des Verweises auf den unterbesetzten Knoten eingefügt. Der damit aus dem Vaterknoten verdrängte Eintrag wird in den unterbesetzten Blattknoten eingefügt [DB06].

Besitzen beide Nachbarknoten des unterbesetzten Blattknotens genau *k* Einträge, so muss einer der Nachbarknoten mit dem Blattknoten verschmolzen werden. Der entstehende Knoten enthält die Einträge der verschmolzenen Knoten sowie den dazugehörigen Eintrag des Vaterknotens. Hierbei kann es zu einer Unterbesetzung des Vaterknotens kommen, die wiederum durch einen Ausgleich oder eine Verschmelzung mit einem Nachbarknoten gelöst werden muss, bis schließlich jeder Knoten mindestens *k* Einträge besitzt und die Löschoperation damit abgeschlossen ist [DB06].

Als erstes Beispiel wird aus dem B-Baum in Abbildung 6 der Eintrag mit Schlüssel 17 gelöscht. Es kommt zu einer Unterbesetzung in Knoten G, die durch Verschmelzung mit Knoten H ausgeglichen wird. Hierbei entsteht ein Knoten mit den Schlüsseln 14, 15, 16 und 18. Außerdem ist nun Knoten C mit nur noch einem Eintrag unterbesetzt, weshalb wieder eine Verschmelzung mit Knoten B durchgeführt wird. Der neu entstehende Knoten enthält die Schlüssel 3, 7, 10 und 13. Damit haben alle Knoten mindestens 2 Einträge und die Löschoperation ist beendet. Es ist der B-Baum aus Abbildung 7 entstanden.

Im zweiten Beispiel wird der Eintrag mit Schlüssel 10 des B-Baumes aus Abbildung 7 entfernt. Da dieser Schlüssel in einem inneren Knoten liegt, wird dieser zunächst mit dem nächstgrößeren[20], nämlich 11, ersetzt. Nun ist Knoten I unterbesetzt. Da der Nachbarknoten H mehr als *k* Einträge besitzt, kann mit Schlüssel 14 ausgeglichen werden. Schlüssel 14 wird deshalb an Stelle der 13 im Vaterknoten eingefügt, Schlüssel 13 in den Knoten I verschoben. Es entsteht der Beispielbaum aus Abbildung 8.

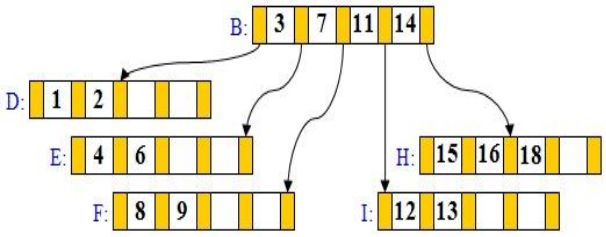

Abbildung 8: Beispielbaum nach dem Löschen der 17 und der 10 (Grad k = 2)

Nachdem nun auch die Operationen eines B-Baumes erklärt wurden, bleibt die Frage zu klären, warum B-Bäume so häufig verwendet werden.

---

[19] Nachbarknoten: Knoten, der unmittelbar links oder rechts des eigenen Verweises im Vaterknoten adressiert wird. Beispiel zu Abbildung 8: F und H sind Nachbarknoten von I, D und E sind keine Nachbarknoten von I.
[20] Man hätte ebenso den nächstkleineren Eintrag wählen können, wodurch aber ein anderer B-Baum entsteht.

4 Schluss

Wie bereits mehrmals bemerkt, werden B-Bäume bis heute in einer Vielzahl von modernen und populären Datenbanksystemen verwendet - beispielsweise in Oracle, IBM DB2, Sybase, Microsoft SQL Server und Informix [LT03]. Selbst Filesysteme wie Windows, das Filesystem von Google oder eines iPhones nutzen inzwischen die Datenstruktur der B-Bäume [IN10].

Die Ursache liegt in ihrer Effizienz, denn aufgrund der selbstständigen Ausbalancierung und der damit verbundenen einheitlichen Tiefe des Baumes kann im Gegensatz zu einem Binärbaum die Durchführung jeder Basisoperation in logarithmischer Zeit[21] garantiert werden [LT03].

Außerdem sind Binärbäume nur für den Hauptspeicher konzipiert. Bei größeren Datenmengen wird jedoch der Hintergrundspeicher benötigt, der von B-Bäumen effizient ausgenutzt werden kann: Jeder Baumknoten kann auf einer Seite[22] gespeichert werden, so dass ein Zugriff auf eine andere Seite nur bei einem Knotenwechsel notwendig wird. Da heutzutage in der Praxis eingesetzte B-Bäume üblicherweise einen Verzweigungsgrad von mindestens 1000 besitzen, genügen beispielsweise 3 Seitenzugriffe, um einen Eintrag unter einem Datensatz mit circa einer Milliarde Einträgen zu finden [DB06][IN10].

Die Tatsache, dass B-Bäume bis heute immer wieder eingesetzt werden, zeigt darüber hinaus auch den Stellenwert der Erfindung Rudolf Bayers, die er bereits 1969 machte.

Nachdem nun der Aufbau und die Funktionalität der B-Bäume in dieser Arbeit geklärt wurden, bleibt nur noch eine Frage: Wofür steht das „B“ in den B-Bäumen?

Rudolf Bayer antwortet nicht ohne ein Grinsen: „Das ist eine der ältesten offenen Fragen der Informatik. Ich werde sie auch nicht beantworten, aus einem ganz einfachen Grund: Das B soll Ihre Fantasie beflügeln. Es hat viel mehr Möglichkeiten in der Interpretation als Sie zunächst vielleicht denken und je mehr Sie darüber nachdenken, was es bedeuten könnte, umso mehr lernen Sie über die Schönheit von B-Bäumen“ [SP02].

Literaturverzeichnis

[CV09] Bayer, Rudolf: *Curriculum Vitae*. 2009.

[DB06] Kemper, Alfons; Eickler, André: *Datenbanksysteme*. Oldenbourg Wissenschaftsverlag, 2006, S. 211-215.

[FS07] Bauer, Friedrich L.: *40 Jahre Informatik in München 1967-2007 - Festschrift.* 2007, S. 191-192.

[IN09] Regler, Alexander: *Interview mit Rudolf Bayer*. 2009.

[21] Bei einer Operation in logarithmischer Zeit müssen im schlimmsten Fall nur logarithmisch viele Baumelemente betrachtet bzw. verschoben werden.

[22] Der Speicher des Hintergrundspeichers ist unterteilt in so genannte Seiten. Eine Seite muss bei einem Seitenzugriff in den Hauptspeicher geladen werden (falls sie sich noch nicht dort befindet), wodurch die Zugriffsdauer verzögert wird.

[IN10] Regler, Alexander: *Interview mit Rudolf Bayer*. 2010.
[SP02] Broy, Manfred; Denert, Ernst: *Software Pioneers - Contributions to Software Engineers.* Springer-Verlag, Berlin, 2002, S. 232-262.

[DD06] http://old.hki.uni-koeln.de/teach/ss06/hs/tag5/DeduktiveDB.pdf, letzter Zugriff am 12.12.2009
[DK05] http://www.inf.ethz.ch/news/spotlight/dist_talks/knuth_DE, letzter Zugriff am 11.12.2009
[DS01] http://drehscheibe.in.tum.de/mitteilungen/auszeichnungen/01/bayer_award.html, letzter Zugriff am 12.12.2009
[DS05] http://drehscheibe.in.tum.de/mitteilungen/auszeichnungen/05/gi_fellow.html, letzter Zugriff am 12.12.2009
[DS99] http://drehscheibe.in.tum.de/mitteilungen/auszeichnungen/99/bayer_orden.html, letzter Zugriff am 12.12.2009
[DW09] http://www.itwissen.info/definition/lexikon/Data-Warehouse-DW-data-warehouse.html, letzter Zugriff am 12.12.2009
[GI05] http://www.gi-ev.de/fileadmin/redaktion/Wettbewerbe/Fellowship/fellow-bayer.pdf, letzter Zugriff am 12.12.2009
[JF05] http://www.juraforum.de/jura/news/news/p/1/id/49691/f/196/, letzter Zugriff am 12.12.2009
[LE03] http://ebus.informatik.uni-leipzig.de/www/media/lehre/seminar-pioniere04/sem04swp-reusche-ausarbeitung.pdf, letzter Zugriff am 28.11.2009
[LT03] http://www.ltrebing.de/studium/02ws/b-baeume/, letzter Zugriff am 30.12.2009
[MC08] http://www.mccreight.com/people/ed_mcc/index.htm, letzter Zugriff am 11.12.2009
[OD09] http://www.bullhost.de/o/objektorientierte-datenbank.html, letzter Zugriff am 12.12.2009
[PU08] http://wwwbayer.in.tum.de/cgi-webcon/webcon/lehrstuhldb/details/Mitarbeiter/num/5/1, letzter Zugriff am 12.12.2009
[SI01] http://www.sigmod.org/sigmodinfo/awards/#innovations, letzter Zugriff am 11.12.2009
[WS09] http://de.mimi.hu/gis/wissensbasiertes_system.html, letzter Zugriff am 12.12.2009