Algorithmik

Sebastian Iwanowski
FH Wedel

3. Lösungen für das Wörterbuchproblem
3.3 Andere Methoden mit Suchbäumen

Algorithmik 3

3.3 Andere Methoden mit Suchbäumen

Ein Dictionary ist eine Datenstruktur für mit einem Schlüssel vergleichbare Elemente,
welche die Funktionen member (key), insert (key, newdata) und delete (key) zur Verfügung stellt

a-b-Baum: Verallgemeinerung von 2-3-Bäumen

Datentyp: Suchbaum mit folgenden Eigenschaften:

i. Alle Daten sind in den Blättern, welche die gleiche Tiefe haben.
ii. Alle inneren Knoten außer der Wurzel haben mindestens a und höchstens b Kinder.
Die Wurzel hat mindestens 2 Kinder.
iii. In den inneren Knoten stehen die jeweils größten Schlüssel aller Daten,
die in den Kinderbäumen enthalten sind.

Alle 3 Dictionary-Funktionen	Laufzeit $\Theta(\log n)$ w.c. und a.c.
	Beweis einfach

Referenzen zum Nacharbeiten und Vertiefen:

Skript Alt S. 44 – 51 (Kap. 3.1.5)

Algorithmik 3

3.3 Andere Methoden mit Suchbäumen

Ein Dictionary ist eine Datenstruktur für mit einem Schlüssel vergleichbare Elemente,
welche die Funktionen member (key), insert (key, newdata) und delete (key) zur Verfügung stellt

AVL-Baum

Datentyp: Binärer Suchbaum mit folgender Eigenschaft:

Für jeden Knoten unterscheidet sich die Höhe der beiden Kindbäume um maximal 1.

Der Erhalt der geforderten Invariante wird durch Rotationen / Doppelrotationen
von Knoten gewährleistet,
welche beim insert bzw. Delete in konstanter Zeit pro Knoten ausgeführt werden.

Alle 3 Dictionary-Funktionen	Laufzeit $\Theta(\log n)$ w.c. und a.c.
	Beweis kompliziert (über Fibonacci-Abschätzung)

Referenzen zum Nacharbeiten und Vertiefen:

Skript Alt S. 41 – 44 (Kap. 3.1.4)	Genaue Funktionsweise und Laufzeitabschätzung darin enthalten, aber nicht prüfungsrelevant

Algorithmik 3

3.3 Andere Methoden mit Suchbäumen

Ein Dictionary ist eine Datenstruktur für mit einem Schlüssel vergleichbare Elemente,
welche die Funktionen member (key), insert (key, newdata) und delete (key) zur Verfügung stellt

Rot-Schwarz-Baum

Datentyp: Binärer Suchbaum, in dem jeder Knoten rot oder schwarz ist, mit folgenden Eigenschaften:

i. Die Wurzel und jedes Blatt sind schwarz.
ii. Rote Knoten haben schwarze Kinder.
iii. Jeder Weg von der Wurzel zum Blatt hat dieselbe Anzahl von schwarzen Knoten.

Der Erhalt der geforderten Invarianten wird durch Rotationen / Doppelrotationen
von Knoten gewährleistet,
welche beim insert bzw. delete in konstanter Zeit pro Knoten ausgeführt werden.

Alle 3 Dictionary-Funktionen Laufzeit $\Theta(\log n)$ w.c. und a.c.

Beweis nachvollziehbar (in Vorlesung)

Referenzen zum Nacharbeiten und Vertiefen:

Skript Alt S. 54 (Kap. 3.1.7)
Cormen Kap. 13 (inkl. Laufzeitabschätzung)

Algorithmik 3

3.3 Andere Methoden mit Suchbäumen

Ein Dictionary ist eine Datenstruktur für mit einem Schlüssel vergleichbare Elemente,
welche die Funktionen member (key), insert (key, newdata) und delete (key) zur Verfügung stellt

Rot-Schwarz-Baum

Satz: Volle*) Rot-Schwarz-Bäume sind leicht zu 2-4-Bäumen transformierbar und umgekehrt.

➔: Jeder schwarze Knoten wird mit seinen roten Kindern verschmolzen.
Die Kinder des oder der bisherigen roten Kindes und das eventuell vorhandene schwarze Kind des alten Knotens werden die Kinder des neuen Knotens.

➔:
i. Ein Knoten mit 2 Kindern wird ein schwarzer Knoten mit zwei schwarzen Kindern.
ii. Ein Knoten mit 3 Kindern a_1, a_2, a_3 wird ein schwarzer Knoten mit schwarzem Kind a_1 und einem neuen roten Kind, das a_2, a_3 als schwarze Kinder hat.
iii. Ein Knoten mit 4 Kindern wird ein schwarzer Knoten mit zwei neuen roten Kindern, die jeweils zwei der bisherigen Kinder als schwarze Kinder haben.

*) Ein Binärbaum heißt voll (oder auch: saturiert), wenn alle inneren Knoten genau 2 Kinder haben.
Ein Binärbaum heißt vollständig, wenn alle Blätter auf den beiden untersten Niveaus liegen (von links nach rechts vollständig aufgefüllt)
Ein Binärbaum heißt perfekt, wenn alle Blätter auf demselben Niveau liegen.

Referenzen zum Nacharbeiten und Vertiefen: Skript Alt S. 54 (Kap. 3.1.7)

Algorithmik 3

3.3 Andere Methoden mit Suchbäumen

Ein Dictionary ist eine Datenstruktur für mit einem Schlüssel vergleichbare Elemente,
welche die Funktionen member (key), insert (key, newdata) und delete (key) zur Verfügung stellt

B-Bäume: Spezielle (a,b)-Bäume

Ein B-Baum ist äquivalent zu einem (b/2,b)-Baum “mindestens zur Hälfte gefüllt”

Ein B*-Baum ist äquivalent zu einem (2b/3,b)-Baum “mindestens zu 2/3 gefüllt”

Die untere Kinderzahl muss mindestens 2 sein!

Anwendung in der Praxis: Festplattenzugriffe

Alle 3 Dictionary-Funktionen Laufzeit Θ(log n) w.c. und a.c.

Referenzen zum Vertiefen:

Cormen Kap. 18

Algorithmik 3

3.3 Andere Methoden mit Suchbäumen

Ein Dictionary ist eine Datenstruktur für mit einem Schlüssel vergleichbare Elemente,
welche die Funktionen member (key), insert (key, newdata) und delete (key) zur Verfügung stellt

Trie-Baum (von Re**trie**val) für Strings über einem k-elementigen Alphabet

Datentyp: k-ärer Suchbaum mit folgenden Eigenschaften:

i. Die Wurzel ist ein leerer Knoten.
ii. Jeder Knoten enthält einen Buchstaben, der in einem String vorkommt.
Für jedes Wort, das diesen Knoten benutzt, gibt es einen Kindknoten
mit dem jeweils nächsten Buchstaben des Strings.
iii. Jeder Knoten hat eine Marke, die anzeigt, ob hier ein Wort endet oder nicht.

Alle 3 Dictionary-Funktionen Laufzeit: Θ(Länge des Strings) w.c. und a.c. (klar)
Laufzeit für Suche in Baum mit n Strings: $\log_k(n)$ a.c. (Knuth, summary)
Speicherplatz für n Strings: n/ln k a.c. (Knuth, summary)

Referenzen zum Nacharbeiten und Vertiefen:

Skript Alt S. 54 – 56 (Kap. 3.1.8) mit verschiedenen Verbesserungsmöglichkeiten,
Knuth ch. 6.3 (Digital Searching) alle nicht prüfungsrelevant