RWTH Aachen
Lehrgebiet Theoretische Informatik
Rossmanith–Hensel–Kuinke–Sánchez

SS 2016
Lösungsvorschlag 3
2. Mai 2016

Übung zur Vorlesung Datenstrukturen und Algorithmen

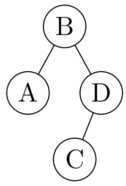
Aufgabe T8

Die Tabellen sehen wie folgt aus:

$w_{i,j}$	A	B	C	D
A	0.2	0.5	0.6	1.0
B	0.0	0.3	0.4	0.8
C	0.0	0.0	0.1	0.5
D	0.0	0.0	0.0	0.4

$e_{i,j}$	A	B	C	D
A	0.2(A)	0.7(B)	0.9(B)	1.8(B)
B	0.0	0.3(B)	0.5(B)	1.3(D)
C	0.0	0.0	0.1(C)	0.6(D)
D	0.0	0.0	0.0	0.4(D)

Daraus entsteht folgender optimaler Suchbaum.



Genauere Erklärung: Die erste Tabelle enthält in Zeile i, Spalte j den Wert für $w_{i,j}$, was als $\sum_{k=i}^{j} p_k$ in der Vorlesung definiert wurde. Das heißt, ein Baum, der die Schlüssel i bis j enthält, wird im Erwartungswert bei einer Suche nach einem zufälligen Schlüssel $w_{i,j}$ mal besucht.

In der Vorlesung wurde präsentiert, wie man die Werte dieser ersten Tabelle effizienter mit Hilfe von dynamischer Programmierung berechnen kann. Hier wird einfach nur ausgenutzt, daß wenn man den Wert für $w_{i,j-1}$ hat, $w_{i,j} = w_{i,j-1} + p_j$ ist. Dies folgt direkt aus der Definition von $w_{i,j}$.

Wir wollen jetzt mit Hilfe dieser Tabelle einen optimalen Suchbaum berechnen. Wir definieren $e_{i,j}$ als den Erwartungswert der Anzahl der Vergleiche in einem optimalen Suchbaum, der die Schlüssel von i bis j enthält. Sei n die Anzahl der Schlüssel. Wir wollen den Baum finden, für den $e_{1,n}$ minimal ist, da $e_{1,n}$ die Anzahl der erwarteten Vergleiche in einem Baum, der alle Schlüssel enthält, ist.

Es gilt $e_{i,j} = \min_{i \le r \le j}(e_{i,r-1} + e_{r+1,j}) + w_{i,j}$: Die Summe $(e_{i,r-1} + e_{r+1,j}) + w_{i,j}$ drückt für ein bestimmtes r aus, daß die erwartete Anzahl an Vergleichen für einen optimalen Suchbaum, der die Schlüssel von i bis j enthält, *wobei der Schlüssel r der Wurzelknoten ist*, die Summe folgender Teile ist:

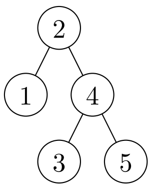
- erwartete Anzahl für die Besuche des Wurzelknotens r ($w_{i,j}$)
- optimale erwartete Anzahl an Vergleichen im linken Unterbaum ($e_{i,r-1}$)
- optimale erwartete Anzahl an Vergleichen im rechten Unterbaum ($e_{r+1,j}$)

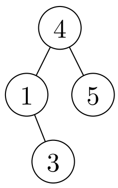
Es ist wichtig hier anzumerken, daß wenn $j < i$ ist, $e_{i,j} = 0$ ist, da dies ein leerer Unterbaum ist. Wenn wir das Minimum über alle möglichen Wurzelknoten wählen, ist das Ergebnis der optimale Wert $e_{i,j}$.

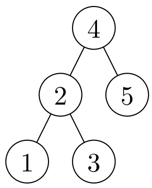
Wir berechnen $e_{1,n}$ wie folgt: Der optimale Suchbaum mit einem einzigen Knoten ist der Knoten selbst. Es gilt also, daß $e_{i,i} = w_{i,i}$, da ein Baum mit einem Knoten im Erwartungswert genau so oft besucht wird wie der Knoten im Erwartungswert besucht wird. Jetzt können wir mit Hilfe der Formel $e_{i,j} = \min_{i \le r \le j}(e_{i,r-1} + e_{r+1,j}) + w_{i,j}$ alle Werte berechnen, wenn $|i - j| = 2$ ist, also für Bäume, die zwei Schlüssel enthalten, da wir alle Werte $e_{i,j}$ kennen, wenn $i = j$ ist. Wir können weiterhin die Werte für Bäume mit immer mehr Knoten berechnen, da wir in der Formel nur die Werte für Bäume brauchen, die weniger Schlüssel enthalten. Jedesmal können wir auch in der Tabelle vermerken, welchen Knoten wir als Wurzelknoten gewählt haben, als wir das Minimum genommen haben. Letztendlich werden wir den optimalen Wert für alle Schlüssel $e_{1,n}$ berechnen.

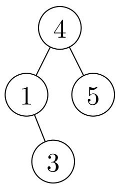
Wir können jetzt aus der Tabelle den optimalen Suchbaum auslesen: In der Zelle für den Wert $e_{1,n}$ steht die Wurzel r des Baums. Da er ein Suchbaum ist, wissen wir, daß der linke Unterbaum die Schlüssel von 1 bis $r - 1$ enthält, und der rechte Unterbaum die Schlüssel von $r+1$ bis n. In der Tabelle finden wir dann in der Zelle für $e_{1,r-1}$ die Wurzel vom linken Unterbaum und in $e_{r+1,n}$ die Wurzel vom rechten Unterbaum. Wir können nach dieser Logik dann den Baum rekursiv aufbauen, indem wir immer in der Tabelle nachgucken, was die nächsten Wurzeln sind. Das Ergebnis ist dann der optimale Suchbaum.

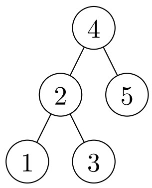
Aufgabe T9

a) 

b) 

c) 

d) 

e) 

Aufgabe T10

Wir beweisen per Induktion über die Struktur eines Binärbaums, daß ein beliebiger Binärbaum nur mit Rotationen in eine verkettete Liste von rechten (bzw. in eine verkettete Liste von linken) Kindern überführt werden kann. Wenn wir dann den Baum t_1 in einen Baum t_2 mit gleichen Schlüsseln überführen wollen, können wir t_1 zunächst in eine Liste von rechten Kindern überführen. Da t_2 in die gleiche Liste überführt werden kann, kann diese Liste auch (mit inversen Rotationen) in t_2 überführt werden.

Induktionsanfang: Wenn wir nur einen Knoten haben, sind keine Rotationen nötig, da die Bäume bereits identisch sind.

Induktionsschritt: Wir beweisen die Aussage für einen Baum mit Wurzel w, linkem Unterbaum t_l mit n_l Knoten und rechtem Unterbaum t_r mit n_r Knoten unter der Voraussetzung, daß die Aussage für t_l und t_r gilt. Wir überführen t_l in eine verkettete Liste von linken Kindern und t_r in eine verkettete Liste von rechten Kindern (Induktionsvoraussetzung). Nun können wir den Baum in eine verkettete Liste von rechten Kindern überführen, indem wir n_l-mal um die Wurzel nach rechts rotieren. Analog können wir den Baum in eine verkettete Liste von linken Kindern überführen, indem wir n_r-mal um die Wurzel nach links rotieren.

Aufgabe H8

Die Tabellen sehen wie folgt aus:

$w_{i,j}$	ETH	KIT	MIT	RWTH	TUM
ETH	0.25	0.30	0.65	0.85	1.00
KIT	0.0	0.05	0.40	0.60	0.75
MIT	0.0	0.0	0.35	0.55	0.70
RWTH	0.0	0.0	0.0	0.20	0.35
TUM	0.0	0.0	0.0	0.0	0.15

$e_{i,j}$	ETH	KIT	MIT	RWTH	TUM
ETH	0.25(ETH)	0.35(ETH)	1.00(MIT)	1.40(MIT)	1.85(MIT)
KIT	0.0	0.05(KIT)	0.45(MIT)	0.85(MIT)	1.30(MIT)
MIT	0.0	0.0	0.35(MIT)	0.75(MIT)	1.20(MIT)
RWTH	0.0	0.0	0.0	0.20(RWTH)	0.50(RWTH)
TUM	0.0	0.0	0.0	0.0	0.15(TUM)

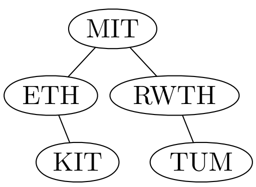
Für $i \leq j$ berechnen wir die einzelnen Werte mit der Formel

$$e_{i,j} = \min_{i \leq r \leq j} (e_{i,r-1} + e_{r+1,j}) + w_{i,j}.$$

In den Klammern hinter den Werten steht, welches r jeweils gewählt wurde, um diesen Wert zu erreichen. Beispielsweise gilt für $e_{\text{ETH,MIT}}$:

$$\begin{aligned}
e_{\text{ETH,MIT}} &= \min_{\text{ETH} \leq r \leq \text{MIT}} (e_{\text{ETH},r-1} + e_{r+1,\text{MIT}}) + w_{\text{ETH,MIT}} \\
&= \min(e_{\text{ETH,ETH-1}} + e_{\text{KIT,MIT}}, e_{\text{ETH,ETH}} + e_{\text{MIT,MIT}}, e_{\text{ETH,KIT}} + e_{\text{RWTH,MIT}}) + 0.65 \\
&= \min(0 + 0.45, 0.25 + 0.35, 0.35 + 0) + 0.65 \\
&= 0.35 + 0.65
\end{aligned}$$

Hier wurde $r = \text{MIT}$ gewählt, um den minimalen Wert 1.70 zu erhalten, also muss die Wurzel des Teilbaums mit allen Schlüsseln zwischen ETH und MIT (also den Schlüsseln ETH, KIT und MIT) der Knoten mit dem Schlüssel MIT sein. Da wir für $e_{\text{ETH,KIT}}$ $r = \text{ETH}$ gewählt haben, ist ETH die Wurzel des Teilbaums mit den Schlüsseln ETH und KIT. Zusammen mit der Bedingung, daß der Baum die Suchbaumeigenschaft erfüllt, ergibt sich eine eindeutige Anordnung des Teilbaums mit den Schlüsseln ETH, KIT und MIT. Ebenso verfahren wir mit den Teilbäumen, die die weiteren Schlüssel RWTH und TUM enthalten. So entsteht folgender optimaler Suchbaum.



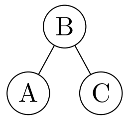
Aufgabe H9

Für die Schlüssel A, B und C mit Zugriffswahrscheinlichkeiten 0.35, 0.3 und 0.35 ergeben sich folgende Tabellen:

$w_{i,j}$	A	B	C
A	0.35	0.65	1.00
B	0.00	0.30	0.65
C	0.00	0.00	0.35

$e_{i,j}$	A	B	C
A	0.35(A)	0.95(A)	1.70(B)
B	0.00	0.30(B)	0.95(C)
C	0.00	0.00	0.35(C)

Daraus entsteht folgender optimaler Suchbaum.



Da die Wurzel mit 0.3 die kleinste Zugriffswahrscheinlichkeit hat, ist dieser Baum ein Gegenbeispiel zu Bobs Behauptung. Also hat Alice Recht.

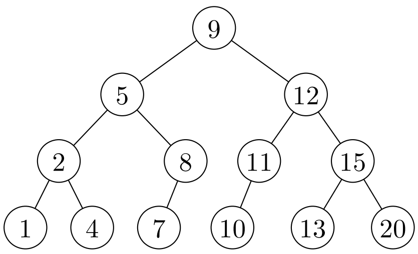
Aufgabe H10

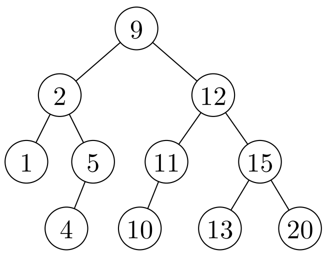
a) Am einfachsten ist es einzusehen, daß die Schlüssel „schichtenweise" von oben nach unten eingefügt worden sein können. Dabei finden gar keine Rotationen statt, da der Baum immer balanziert bleibt. Diese Reihenfolge ist, wenn wir von links nach rechts gehen:

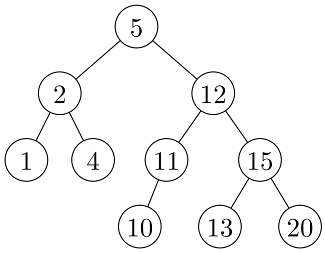
$$9, 5, 15, 2, 8, 11, 20, 1, 4, 7, 10, 12$$

Es gibt natürlich viele andere Möglichkeiten. Allein wenn wir schichtenweise vorgehen, gibt es $2! \cdot 4! \cdot 5! = 5760$ Möglichkeiten.

Insgesamt gibt es aber noch viel mehr Reihenfolgen, nämlich genau 7,076,160.

b) 

c) 

d) 

e)