Humboldt-Universität zu Berlin
Institut für Informatik
Prof. Dr. Marius Kloft
M. Bux, B. Grußien, P. Schäfer, F. Tschorsch, K. Völlinger

Berlin, den 11.07.2016

Übung zur Vorlesung
Algorithmen und Datenstrukturen

Beispiel zu Optimalen Suchbäumen

Aufgabe

Gegeben seien die Schlüsselmenge $S = \{10, 25, 30, 50\}$ und die folgenden Zugriffshäufigkeiten:

i	0	1	2	3	4
k_i		10	25	30	50
a_i		1	2	0	3
b_i	1	2	1	0	0

Konstruieren Sie mit Hilfe des auf dynamischer Programmierung basierenden Algorithmus aus der Vorlesung den optimalen Suchbaum für die obigen Zugriffshäufigkeiten.

Lösung

$T_{i,j}$ ist der optimale Suchbaum für $\{k_{i+1}, \dots, k_j\}$. $T_{i,j}$ enthält (in seiner erweiterten Form) die Knoten

$$]k_i, k_{i+1}[,\ k_{i+1},\ ]k_{i+1}, k_{i+2}[,\ k_{i+2},\ \dots,\ k_j,\ ]k_j, k_{j+1}[.$$

Im Folgenden wird der Algorithmus zum Bestimmen der Kosten $P(T_{i,j})$ der Teilbäume $T_{i,j}$ für $i \in \{0, \dots, n\}$, $j \in \{i, \dots, n\}$ angegeben. Hierbei ist $P(i,j) = P(T_{i,j})$ und $W(i,j) = W(T_{i,j})$.

for $i = 0$ **to** n **do**
    $W(i,i) := b_i$;
    $P(i,i) := b_i$;
end for
for $b = 1$ **to** n **do**
    for $i = 0$ **to** $n - b$ **do**
        $j := i + b$;
        $W(i,j) := W(i,j-1) + a_j + b_j$;
        Bestimme $l \in \{i+1, \dots, j\}$ so dass $P(i,l-1) + P(l,j)$ minimal ist;
        $P(i,j) := W(i,j) + P(i,l-1) + P(l,j)$;
    end for
end for

Nun seien

$$\begin{aligned} &l_{i,j} \text{ der Wert } l \text{ für den } P(i,l-1) + P(l,j) \text{ minimal ist,} \\ P_{i,j} &:= P(i,j) = P(T_{i,j}) \quad \text{und} \\ W_{i,j} &:= W(i,j) = W(T_{i,j}). \end{aligned}$$

$i \setminus j$	0	1	2	3	4
0	$W_{0,0} = b_0 = 1$ $P_{0,0} = b_0 = 1$	$l_{0,1} = 1$ $W_{0,1} = 4$ $P_{0,1} = 4 + 1 + 2 = 7$	$l_{0,2} = 2$ $W_{0,2} = 7$ $P_{0,2} = 7 + 7 + 1 = 15$	$l_{0,3} = 2$ $W_{0,3} = 7$ $P_{0,3} = 7 + 7 + 2 = 16$	$l_{0,4} = 2$ $W_{0,4} = 10$ $P_{0,4} = 10 + 7 + 6 = 23$
1		$W_{1,1} = b_1 = 2$ $P_{1,1} = b_1 = 2$	$l_{1,2} = 2$ $W_{1,2} = 5$ $P_{1,2} = 5 + 2 + 1 = 8$	$l_{1,3} = 2$ $W_{1,3} = 5$ $P_{1,3} = 5 + 2 + 2 = 9$	$l_{1,4} = 2$ $W_{1,4} = 8$ $P_{1,4} = 8 + 2 + 6 = 16$
2			$W_{2,2} = b_2 = 1$ $P_{2,2} = b_2 = 1$	$l_{2,3} = 3$ $W_{2,3} = 1$ $P_{2,3} = 1 + 1 + 0 = 2$	$l_{2,4} = 4$ $W_{2,4} = 4$ $P_{2,4} = 4 + 2 + 0 = 6$
3				$W_{3,3} = b_3 = 0$ $P_{3,3} = b_3 = 0$	$l_{3,4} = 4$ $W_{3,4} = 3$ $P_{3,4} = 3 + 0 + 0 = 3$
4					$W_{4,4} = b_4 = 0$ $P_{4,4} = b_4 = 0$

Ermitteln des optimalen (nicht erweiterten) Suchbaums:

Index $l_{i,j}$ gibt uns den Index des Schlüssels an, der in der Wurzel von $T_{i,j}$ steht. Weiterhin hängt links an der Wurzel der Teilbaum $T_{i,l_{i,j}-1}$ und rechts der Teilbaum $T_{l_{i,j},j}$. Daraus lässt sich rekursiv der Suchbaum bestimmen.

Der berechnete optimale Suchbaum für das Beispiel ergibt sich wiefolgt:
Der optimale Suchbaum ist $T_{0,4}$. Index $l_{0,4} = 2$ ist der Index des Wurzelknotens, d.h. $k_2 = 25$ ist der Wurzelknoten. Links an k_2 hängt der Teilbaum $T_{0,1}$, rechts der Teilbaum $T_{2,4}$.
Der Teilbaum $T_{0,1}$ hat $k_1 = 10$ als Wurzel, da $l_{0,1} = 1$ ist, und keine Kinder (links hängt der Teilbaum $T_{0,0}$, rechts der Teilbaum $T_{1,1}$, beide enthalten keine Schlüssel).
Der Teilbaum $T_{2,4}$ hat $k_4 = 50$ als Wurzel, da $l_{2,4} = 4$ ist. Links hängt der Teilbaum $T_{2,3}$, rechts der leere Teilbaum $T_{4,4}$.
$T_{2,3}$ hat $k_3 = 30$ als Wurzel, da $l_{2,3} = 3$ ist, und keine Kinder.