Eidgenössische
Technische Hochschule
Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

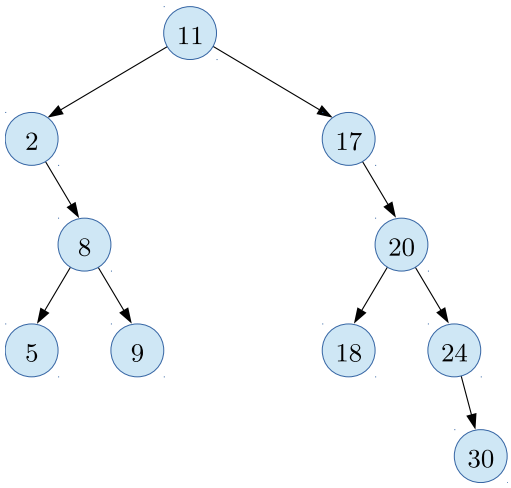
Department of Computer Science
Markus Püschel
Peter Widmayer
Thomas Tschager
Tobias Pröger

1st December 2016

Algorithms & Data Structures — Solutions to Sheet 10 — AS 16

Solution 10.1 *Search Trees.*

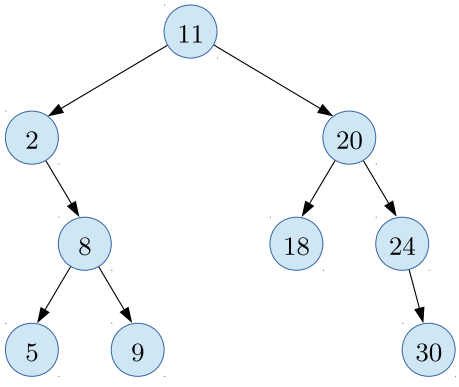
a) Resulting tree:



For the sake of clarity, the leaves (the null-successors) are not shown in the figure.

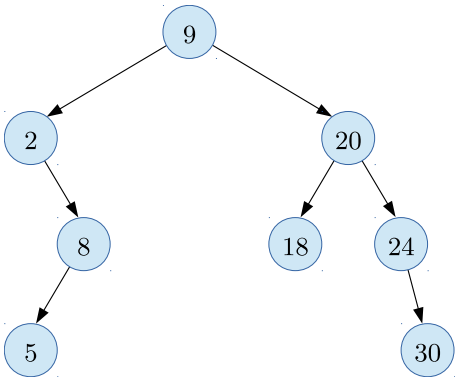
b)
- Preorder: 11, 2, 8, 5, 9, 17, 20, 18, 24, 30,
- Postorder: 5, 9, 8, 2, 18, 30, 24, 20, 17, 11,
- Inorder: 2, 5, 8, 9, 11, 17, 18, 20, 24, 30.

c) If we remove the key 17, the node storing the key 20 becomes a successor of the root:

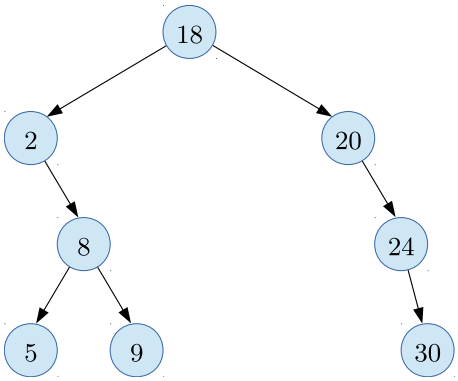


To remove the key 11, we first replace it with the key of the *symmetric predecessor* (i.e. by the largest key smaller than 11) or with the key of the *symmetric successor* (i.e. by the smallest key larger than 11) and then remove the corresponding predecessor or successor.

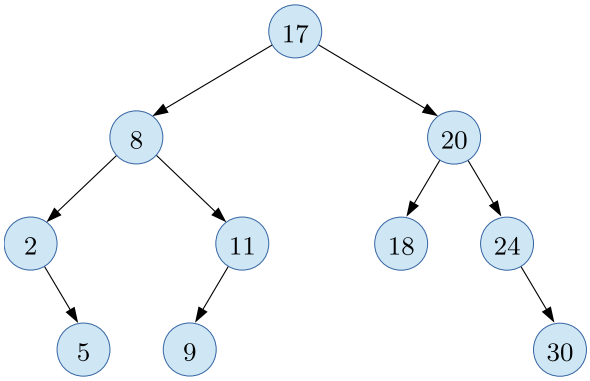
Using the symmetric predecessor:



Using the symmetric successor:



d) We obtain the following tree:



Solution 10.2 *Questions on Search Trees.*

a) The node w is in the subtree of v if the following properties hold: v has a smaller preorder number and a larger postorder number than w. Let us first consider the preorder numbers: A node a is numbered before all of his successors and, therefore, has a smaller preorder number. However, a smaller preoder number is not a sufficient for being sure that a node is in the subtree of a, because there could also be nodes in other subtrees that are numbered after all nodes in the subtree of a. Analogously, it holds that the postorder number of a is larger than the postorder number of every node in the subtree of a. This also holds for nodes in other subtrees that are numbered before a. However, a node has both properties (v has a smaller preoder number and a larger postorder number than w) if and only if the node is in the subtree of a.

b) Only the insertion of one of the keys 8, 10, or 11 leads to a double rotation.

Solution 10.3 *Number of different Search Trees.*

Let $\mathcal{K}_n = \{1, \ldots, n\}$ be a set of keys, and let $T(n)$ be the number of different search trees for the key set $\mathcal{K}_n$. For $n = 0$ we have $\mathcal{K}_0 = \emptyset$ and $T(0) = 1$, because only the empty tree corresponds to $\mathcal{K}_0$. For $n = 1$ we have $\mathcal{K}_1 = \{1\}$, and the only possible search tree contains just the first key, so $T(1) = 1$.

For general n, we proceed as follows. Assume that the key k is stored at the root of a search tree.

Then the left subtree contains the keys $\{1, \ldots, k-1\}$, and analogously the right subtree contains the keys $\{k+1, \ldots, n\}$. There are $T(k-1)$ possible subtrees on the left and $T(n-k)$ possible subtrees on the right. Since every possible subtree on the left can be combined with every possible subtree on the right, the numbers have to be multiplied. Thus, there are $T(k-1) \cdot T(n-k)$ possible search trees for key set $\mathcal{K}_n$ with the key k at the root. Now we only need to sum over all possible keys k and obtain

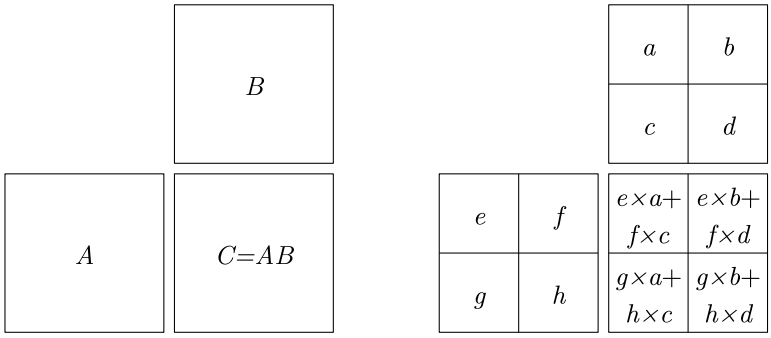
$$T(n) = \sum_{k=1}^{n} T(k-1)T(n-k). \tag{1}$$

Note: These numbers are known under the name *Catalan numbers* (named after the Belgian Mathematician Eugène Charles Catalan). We have (without a proof)

$$T(n) = \frac{1}{n+1}\binom{2n}{n}.$$

Solution 10.4 *Matrix Multiplication.*

a) If we are given two matrices A and B as input and if we should compute the matrix $C = A \times B$, then we have the following situation:



We recursively compute the partial products $P_1 = (f-h) \times (c+d)$, $P_2 = (e-g) \times (a+b)$, $P_3 = (e+h) \times (a+d)$, $P_4 = (e+f) \times d$, $P_5 = (g+h) \times a$, $P_6 = e \times (b-d)$, and $P_7 = h \times (c-a)$. We obtain

- $P_1 + P_3 - P_4 + P_7 = (f-h) \times (c+d) + (e+h) \times (a+d) - (e+f) \times d + h \times (c-a) = f \times c + f \times d - h \times c - h \times d + e \times a + e \times d + h \times a + h \times d - e \times d - f \times d + h \times c - h \times a = e \times a + f \times c$,
- $P_4 + P_6 = (e+f) \times d + e \times (b-d) = e \times d + f \times d + e \times b - e \times d = e \times b + f \times d$,
- $P_5 + P_7 = (g+h) \times a + h \times (c-a) = g \times a + h \times a + h \times c - h \times a = g \times a + h \times c$,
- $P_3 - P_2 + P_6 - P_5 = (e+h) \times (a+d) - (e-g) \times (a+b) + e \times (b-d) - (g+h) \times a = e \times a + e \times d + h \times a + h \times d - e \times a - e \times b + g \times a + g \times b + e \times b - e \times d - g \times a - h \times a = g \times b + h \times d$.

b) The multiplication of two (1×1) matrices needs exactly one elementary operation, that is $A(1) = 1$. If we multiply two $(n \times n)$ matrices, we first have to extract 14 submatrices of size $(n/2 \times n/2)$, namely the factors for the recursive multiplications (i.e. $(f-h), (c+d), (e-g), (a+b), \ldots, e, (b-d), h, (c-a)$). Then, we compute the seven recursive multiplications of two $(n/2 \times n/2)$ matrices. The product matrix C can be obtained with 8 additions, respectively subtractions, of some of these partial products. In every recursive step, we need $\Theta(n^2)$ elementary additions and subtractions if we do not consider the recursive calls

(n is the number of rows and columns in the corresponding matrices). We obtain the recursive equation

$$A(n) = \begin{cases} 1 & \text{if } n = 1, \\ 7A(n/2) + \xi n^2 & \text{if } n > 1, \end{cases} \tag{2}$$

where ξ denotes an appropriately chosen constant. If only additions and subtractions shall be considered, then we can choose $\xi = \frac{18}{4}$. The constant has to be chosen larger if also additional operations such as the memory allocation to store intermediate results shall be considered. By telescoping, we obtain for $n = 2^k$

$$\begin{aligned}
A(n) &= 7A(n/2) + \xi n^2 \\
&= 7\big(7A(n/2^2) + \xi(n/2)^2\big) + \xi n^2 \\
&= 7^2 A(n/2^2) + \xi\big(7^1(n/2)^2 + 7^0 n^2\big) \\
&= 7^2\big(7A(n/2^3) + \xi(n/2^2)^2\big) + \xi\big(7^1(n/2)^2 + 7^0 n^2\big) \\
&= 7^3 A(n/2^3) + \xi\big(7^2(n/2^2)^2 + 7^1(n/2)^2 + 7^0 n^2\big) \\
&= 7^3\big(7A(n/2^4) + \xi(n/2^3)^2\big) + \xi\big(7^2(n/2^2)^2 + 7^1(n/2)^2 + 7^0 n^2\big) \\
&= 7^4 A(n/2^4) + \xi\big(7^3(n/2^3)^2 + 7^2(n/2^2)^2 + 7^1(n/2)^2 + 7^0 n^2\big) \\
&= 7^4 A(n/2^4) + \xi n^2\left(\frac{7^3}{(2^3)^2} + \frac{7^2}{(2^2)^2} + \frac{7^1}{(2^1)^2} + \frac{7^0}{(2^0)^2}\right) \\
&= 7^4 A(n/2^4) + \xi n^2\left(\frac{7^3}{4^3} + \frac{7^2}{4^2} + \frac{7^1}{4^1} + \frac{7^0}{4^0}\right) = \cdots \\
&\overset{(!)}{=} 7^k A(n/2^k) + \xi n^2 \sum_{i=0}^{k-1} \left(\frac{7}{4}\right)^i = 7^{\log_2 n} + \xi n^2 \cdot \frac{(7/4)^{\log_2 n} - 1}{7/4 - 1} \\
&= n^{\log_2 7} + \xi n^2 \cdot \frac{n^{\log_2(7/4)} - 1}{3/4} = n^{\log_2 7} + \frac{4}{3}\xi n^2\left(n^{\log_2 7 - \log_2 4} - 1\right) \\
&= n^{\log_2 7} + \frac{4}{3}\xi n^{\log_2 7} - \frac{4}{3}\xi n^2,
\end{aligned} \tag{3}$$

which we can prove by mathematical induction over n.

Base case ($n = 1$):

$$A(1) = 1 = 1 + \frac{4}{3}\xi - \frac{4}{3}\xi = 1^{\log_2 7} + \frac{4}{3}\cdot\xi\cdot 1^{\log_2 7} - \frac{4}{3}\cdot\xi\cdot 1^2 \tag{4}$$

Induction hypothesis: For some $n \in \mathbb{N}$, assume that $A(n) = n^{\log_2 7} + \frac{4}{3}\xi n^{\log_2 7} - \frac{4}{3}\xi n^2$.

Inductive step ($n \to 2n$):

$$\begin{aligned}
A(2n) &= 7A(n) + \xi(2n)^2 \qquad (5) \\
&\overset{(I.H.)}{=} 7\left(n^{\log_2 7} + \frac{4}{3}\xi n^{\log_2 7} - \frac{4}{3}\xi n^2\right) + 4\xi n^2 \\
&= 7n^{\log_2 7} + 7\cdot\frac{4}{3}\xi n^{\log_2 7} - \frac{28}{3}\xi n^2 + \frac{12}{3}\xi n^2 \\
&= 2^{\log_2 7} n^{\log_2 7} + 2^{\log_2 7}\cdot\frac{4}{3}\xi n^{\log_2 7} - \frac{16}{3}\xi n^2 \\
&= (2n)^{\log_2 7} + \frac{4}{3}\xi(2n)^{\log_2 7} - \frac{4}{3}\xi(2n)^2. \qquad (6)
\end{aligned}$$