Fundamental Algorithms, Assignment 6
Solutions

1. In a BST assume we have a function \( TD[x] \) that gives the number of descendents (including \( x \) itself) of node \( x \). Now we apply \( DELETE[z] \). Assume\(^1\) that \( z \) is childless. Give a procedure that will update the function \( DELETE \) in time \( O(H) \).

**Solution:** Only the ancestors \( w \) of \( z \) will have \( TD[w] \) changes, and their values will be decremented by one. One climbs from \( z \) to the root decrementing \( TD \) at each spot. For example:

\[
\text{\( w = PARENT[z] \)}
\]

\[
\text{WHILE } \text{\( w \neq NIL \)}
\]

\[
\text{T}D(w) --
\]

\[
\text{w} \leftarrow PARENT[w]
\]

**ENDWHILE**

2. Determine an LCS of 10010101 and 010110110.

**Solution:** We create an eight by eight array giving \( C[m,n] \), the length of the LCS between the first \( m \) of the first sequence and the first \( n \) of the second sequence.

Here is array. The sequences are placed on top and on the left for convenience. The numbering starts at 0 so that the row zero and column zero are all zeroes.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>0</th>
<th>1</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>

So the length is 6. Start at the bottom right and walk until hitting the edge. At \( (i,j) \) go diagonal left if \( C[i,j] = C[i-1,j-1] + 1 \); if not go left or up, whichever is \( C[i,j] \). (We’ll go left if they both are.) This gives

\(^1\)Other cases are doable, but harder
The places where you go diagonally left are the same in both sequences and these give the common sequence 010101. Note that there is no uniqueness to the sequences themselves.

3. Write all the parenthesizations of $ABCDE$. Associate them in a natural way with (setting $n = 5$) the terms $P(i)P(5 - i)$, $i = 1, 2, 3, 4$ given in the recursion for $P(n)$.

Solution: Splitting 1 – 4 gives $P(1)P(4) = 5$ parenthesizations:

$$A(B(C(DE))), A(B((CD)E)), A((BC)(DE)), A((B(CD))E), A(((BC)D)E)$$

Splitting 4 – 1 gives $P(4)P(1) = 5$ parenthesizations:


Splitting 2 – 3 gives $P(2)P(3) = 2$ parenthesizations:

$$((AB)((CD)E)), (AB)(C(DE))$$

Splitting 3 – 2 gives $P(3)P(2) = 2$ parenthesizations:

$$((AB)C)(DE), (A(BC))(DE)$$
4. Let \( x_1, \ldots, x_m \) be a sequence of distinct real numbers. For \( 1 \leq i \leq m \) let \( INC[i] \) denote the length of the longest increasing subsequence ending with \( x_i \). Let \( DEC[i] \) denote the length of the longest decreasing subsequence ending with \( x_i \).

(a) Find an efficient method for finding the values \( INC[i], 1 \leq i \leq n \).

Solution: The longest increasing subsequence ending in \( x_i \) is either simply \( x_i \) or it is obtained by appending \( x_i \) to some subsequence ending in \( x_j \) where \( j < i \). One can do that if and only if \( x_j < x_i \). So we should take \( INC[i] \) to be 1 (\( x_i \) itself) plus the maximum of the \( INC[j], j < i \), for which \( x_j < x_i \). However, if there are no such \( j \) (for example, when \( i = 1 \)) the default value should be 1. Each \( INC[i] \) then takes a single loop which is time \( O(n) \) and so the total time is \( O(n^2) \). (Of course, \( DEC[i] \) can be found similarly.)

(b) Let \( LIS \) denote the length of the longest increasing subsequence of \( x_1, \ldots, x_m \). Show how to find \( LIS \) from the values \( INC[i] \).

Similarly, let \( DIS \) denote the length of the longest decreasing subsequence of \( x_1, \ldots, x_m \). Show how to find \( DIS \) from the values \( DEC[i] \).

Solution: \( LIS \) is simply the maximum of all \( INC[i], 1 \leq i \leq n \), as the subsequence has to end somewhere. Similarly, \( DIS \) is simply the maximum of all \( DEC[i], 1 \leq i \leq n \).

(c) Suppose \( i < j \). Prove that it is impossible to have \( INC[i] = INC[j] \) and \( DEC[i] = DEC[j] \).

Solution: Suppose \( x_i < x_j \). Then \( INC[j] \geq INC[i] + 1 \) since you can take the maximal increasing sequence ending at \( x_i \) and append \( x_j \). (That may not be optimal, but \( INC[j] \) is at least that length.)

Similarly, suppose \( x_i > x_j \). Then \( DEC[j] \geq DEC[i] + 1 \) since you can take the maximal decreasing sequence ending at \( x_i \) and append \( x_j \).

(d) Deduce the following celebrated results (called the Monotone Subsequence Theorem) of Paul Erdős and George Szekeres: Let \( m = ab + 1 \). Then any sequence \( x_1, \ldots, x_m \) of distinct real numbers either \( LIS > a \) or \( DIS > b \). (Idea: Assume not and look at the pairs \( (INC[i], DEC[i]) \).)
Solution: If \( LIS \leq a \) and \( DIS \leq b \) then there are only \( ab \) possibilities for the pair \((INC[i], DEC[i])\), but from the previous part we have \( ab + 1 \) distinct pairs!

5. Find an optimal parenthesization of a matrix-chain product whose sequence of dimensions is 5, 10, 3, 12, 5, 50, 6. **Solution:**

The matrix chain product of \( A_1 A_2 A_3 \ldots A_n \) can be broken down to \((A_1 \ldots A_k)(A_{k+1} \ldots A_n)\). To find an optimal parenthesization for \( n \) matrices, we find the subset of \( k \) matrices, where \( k < n \). And then compose them altogether.

In our algorithm, we have two matrices, one to record the minimum number of operations it takes and the other to record the parenthesization.

\[
\text{Matrix}[i][j] = 0 \quad (i = j)
\]
\[
\text{Matrix}[i][j] = \text{min}(\text{Matrix}[i][k] + \text{Matrix}[k+1][j] + \text{size}[i-1] \times \text{size}[k+1] \times \text{size}[i+j])
\]
\[
\text{Result}[i][j] = k + 1 \quad \text{which gives min values to Matrix}[i][j]
\]

**MATRIX-CHAIN-ORDER()**

\[
\text{for}(t = 1; t < p; t++)
\]
\[
\text{for}(i = 0; i < p - t; i++)
\]
\[
\text{for}(k = i; k < i + t; k++)
\]
\[
\text{matrix}[i][i+t] = \text{matrix}[i][k] + \text{matrix}[k+1][i+t] + \text{size}[i] \times \text{size}[k+1] \times \text{size}[i+t]
\]
\[
\text{result}[i][i+t] = k + 1;
\]

Matrix[i][j] as following

\[
\begin{array}{cccccc}
0 & 150 & 330 & 405 & 1655 & 2010 \\
0 & 0 & 360 & 330 & 2430 & 1950 \\
0 & 0 & 0 & 180 & 930 & 1770 \\
0 & 0 & 0 & 0 & 3000 & 1860 \\
0 & 0 & 0 & 0 & 0 & 1500 \\
0 & 0 & 0 & 0 & 0 & 0
\end{array}
\]

Result[i][j] as following

\[
\begin{array}{cccccc}
0 & 150 & 330 & 405 & 1655 & 2010 \\
0 & 0 & 360 & 330 & 2430 & 1950 \\
0 & 0 & 0 & 180 & 930 & 1770 \\
0 & 0 & 0 & 0 & 3000 & 1860 \\
0 & 0 & 0 & 0 & 0 & 1500 \\
0 & 0 & 0 & 0 & 0 & 0
\end{array}
\]
Therefore the optimal parenthesization is \((AB)((CD)(EF))\)

For example, Matrix\([2][5]\) gives the optimal matrix chain product of CDEF. The optimal choice comes from the minimum of C(DEF), (CD)(EF), (CDE)F. Take C(DEF) for example. It divides into subproblem C and DEF. C is given by Matrix\([2][2]\), which is 0 since C is itself. DEF is given by Matrix\([3][5]\), which is 1860. C is a matrix of 3*12. The result of DEF is a matrix of 12*6. Therefore, \(p_{i-1}p_kp_j\) equals 3*12*6 = 216. The number of operations taken to get C(DEF) is therefore 1860 + 216 = 2076. We can also get (CD)(EF) and (CDE)F with the same manner. They are 1770 and 1830. As a result, we take 1770 for Matrix\([2][5]\) and 4 for \(k+1\), which is recorded in Result\([2][5]\).

6. Some exercises in logarithms:

(a) Write \(\log(4^n/\sqrt{n})\) in simplest form. What is its asymptotic value.
Solution: \(n \log(4) - \frac{1}{2} \log(n) = 2n - \frac{\log n}{2}\).

(b) Which is bigger, \(5^{313340}\) or \(7^{271251}\)? Give reason. (You can use a calculator.)
Solution: The numbers themselves are too big for calculators but compare their \(\log\)s, which are around 727000 and 761000 respectively so the second is bigger.

(c) Simplify \(n^2 \log(n^2)\) and \(\log^2(3^n)\).
Solution: \(2n^2 \log(n)\) and \((3 \log n)^2 = 9 \log^2 n\).

(d) Solve (for \(x\)) the equation \(e^{-x^2/2} = \frac{1}{n}\).
Solution: \(-\frac{x^2}{2} = \log(1/n) = -\log n\) so \(\frac{x^2}{2} = \log n\) so \(x^2 = 2 \log n\) so \(x = \sqrt{2 \log n}\).

(e) Write \(\log_n 2^n\) and \(\log_n n^2\) in simple form.
Solution: The first is that \(x\) for which \(n^x = 2^n\) so \(x \log(n) = n\) so \(x = \frac{n}{\log(n)}\) is the answer. For the second the answer is 2.
(f) What is the relationship between \( \log n \) and \( \log_3 n \)?

*Solution*: \( \log_3 n = \frac{\log n}{\log 3} \). As \( \log(3) \sim 1.5 \) is a constant they are “the same” in \( \Theta \)-land.

(g) Assume \( i < n \). How many times need \( i \) be doubled before it reaches (or exceeds) \( n \)?

*Solution*: If we double \( x \) times we reach \( i 2^x \) so we need \( i 2^x \geq n \), or \( 2^x \geq \frac{n}{i} \) or \( x \geq \log(\frac{n}{i}) \). As \( x \) need be an integer the precise number of times is \( \lceil \log(\frac{n}{i}) \rceil \).

(h) Write \( \log[n^n e^{-n} \sqrt{2\pi n}] \) precisely as a sum in simplest form. What is it asymptotic to as \( n \to \infty \)? What is interesting about the bracketed expression?

*Solution*: This is Stirling’s Formula and is asymptotic to \( n! \). Precisely

\[
\log[n^n e^{-n} \sqrt{2\pi n}] = n \log n - n \log e + \frac{1}{2} \log(2\pi) + \frac{1}{2} \log n
\]

which is asymptotic to \( n \log n \).